Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
Факультет інформаційних технологій
Кафедра безпеки інформації та телекомунікацій

ПОЯСНЮВАЛЬНА ЗАПИСКА

кваліфікаційної роботи ступеня магістра

студента Олефіра Кирила Андрійовича

академічної групи 125м-22-1

спеціальності 125 Кібербезпека

спеціалізації[1]

за освітньо-професійною програмою Кібербезпека

на тему Засоби контролю інформаційної безпеки в базах даних SQL

Керівники	Прізвище, ініціали	Оцінка за шкалою		Підпис
		рейтинговою	інституційною	
кваліфікаційної роботи	д. ф.-м. наук, проф. Кагадій Т. С.			
розділів:				
спеціальний	ас. Олішевський І. Г.			
економічний	к. е. н. доц. Пілова Д. П.	90	відмінно	

Рецензент				

Нормоконтролер	ст. викл. Мєшков В. І.			

Дніпро

2023

ЗАТВЕРДЖЕНО:
завідувач кафедри
безпеки інформації та телекомунікацій
_____д.т.н., проф. Корнієнко В.І.

«___»_____20___року

ЗАВДАННЯ

на кваліфікаційну роботу ступеня магістра

студенту ___Олефіру К.А._____ **академічної групи** __125м-22-1_____

спеціальності _____125 Кібербезпека_____

спеціалізації _____

за освітньо-професійною програмою ____Кібербезпека_____

_____

на тему ___Засоби контролю інформаційної безпеки в базах даних SQL_____

_____

Затверджену наказом ректора НТУ «Дніпровська політехніка» від _____№_____

Розділ	**Зміст**	**Термін виконання**
Стан питання. Постановка задачі	Огляд сучасних методів захисту та аналіз ризиків баз даних SQL	Вересень 2023
Спеціальна частина	Розробка процедур та політик для підвищення рівня безпеки баз даних	Жовтень 2023
Економічний розділ	Оцінка економічної ефективності заходів інформаційної безпеки	Листопад 2023

Завдання видано _____ __Тетяна КАГАДІЙ_____
(підпис керівника) (ім'я, прізвище)

Дата видачі завдання: _____

Дата подання до екзаменаційної комісії: _____

Прийнято до виконання _____ __Кирило ОЛЕФІР_____
(підпис студента) (ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 96 с., 5 рис., 2 табл. 4 додатки, 17 джерел.

Об`єкт дослідження: Об'єктом дослідження є засоби контролю інформаційної безпеки в базах даних SQL. Мета роботи полягає в дослідженні особливостей безпеки в сучасних СУБД - MySQL, Oracle, та MS SQL Server.

Мета роботи: дослідити особливості безпеки сучасних СУБД MySQL, Oracle, MS SQL Server.

Методи розробки: спостереження, порівняння, аналіз, опис.

У першому розділі були поставлені задачі висвітлити загрози безпеки у сучасних SQL базах даних та висвітлити можливості та недоліки на прикладі MySQL, Oracle і SQL Server.

Спеціальна частина була присвячена засобам безпеки трьох найбільш поширених на сьогоднішній день СУБД: MySQL, Oracle і SQL Server. Було проведено дослідження стосовно комп'ютерних та некомп'ютерних засобів контролю інформаційної безпеки сервісів MySQL, Oracle і SQL Server.

В економічному вимірі дослідження визначено ефективність використання систем управління базами даних (СУБД) MySQL, Oracle та MS SQL Server та проведено оцінку вартості впровадження та підтримки кожної з розглянутих СУБД, враховуючи економічні вигоди, що можуть виникнути внаслідок покращення безпеки даних.

Наукова новизна полягає у виявленні ефективних стратегій захисту, які враховують актуальні виклики та загрози у цій сфері. Дослідження розкриває нові підходи до забезпечення інформаційної безпеки в базах даних SQL

Практичне значення роботи полягає у застосуванні для компаній та організацій, що працюють з базами даних, сприяючи підвищенню рівня інформаційної безпеки та оптимізації витрат на цей напрямок.

ІНФОРМАЦІЙНА БЕЗПЕКА, СУБД, MYSQL, ORACLE, MS SQL SERVER, ЗАХИСТ ДАНИХ, АУТЕНТИФІКАЦІЯ, АВТОРИЗАЦІЯ, АДМІНІСТРУВАННЯ, РЕЗЕРВНЕ КОПІЮВАННЯ, ШИФРУВАННЯ, АУДИТ.

ABSTRACT

Explanatory note: 96 pages, 5 images, 2 tables, 4 applications, 17 sources.

Object of investigation: Object of investigation is information security control techniques in SQL databases. Meta-works focuses on the investigation of security features in current DBMSs - MySQL, Oracle, and MS SQL Server.

Meta robots: investigate the security features of current DBMS MySQL, Oracle, MS SQL Server.

Methods of investigation: caution, leveling, analysis, description.

The first section was tasked with identifying security threats in current SQL databases and identifying capabilities and shortcomings in MySQL, Oracle and SQL Server applications.

A special part was devoted to the security of the three most advanced DBMSs today: MySQL, Oracle and SQL Server. An investigation was carried out of completely computer and non-computer information security control systems of MySQL, Oracle and SQL Server services.

In the economic world, research has revealed the effectiveness of database management systems (DBMS) MySQL, Oracle and MS SQL Server, and an assessment has been made of the effectiveness of skin care and skin care from the various DBMSs, healthcare and economic And the benefits that may result from the reduction in data security.

Scientific novelty lies in the identified effective strategies to protect against current trends and threats in this area. The research reveals new approaches to ensuring information security in SQL databases

The practical significance of the work lies with the establishment of companies and organizations that work with databases, consistent with the increased level of information security and optimization of costs in this area.

INFORMATION SECURITY, DBMS, MYSQL, ORACLE, MS SQL SERVER, DATA SECURITY, AUTHENTICATION, AUTHORIZATION, ADMINISTRATION, BACKUP, ENCRYPTION, AUDIT.

СПИСОК УМОВНИХ СКОРОЧЕНЬ

БД – База даних;

ОС – Операційна система;

ПЗ – Програмне забезпечення;

СУБД – Система управління базами даних;

ACL – Access Control List;

API – Application programming interface;

COM – Component Object Model;

DMO – Distributed Management Objects;

DNS – Domain Name System;

IDS – Intrusion Detection System;

PSS – Process Status Structure;

SID – Security Identifier;

SIEM – Security information and event management

SQL – Structured Query Language;

SSL – Secure Sockets Layer;

TLS – Transport layer security;

UNIX – Uniplexed Information and Computing Service;

VPD – Virtual Private Database;

ЗМІСТ

ВСТУП

Найвища цінність сучасного світу – інформація. У більшості своїй вона все частіше переноситься і зберігається у вигляді електронних баз даних, що знаходяться під контролем систем управління базами даних (СУБД), які дозволяють кінцевим користувачам або розробникам додатків спільно використовувати дані та керувати ними.

Для того щоб говорити про безпеку інформації, що зберігається і використовується таким чином, необхідно розглянути захищеність всієї інформаційної системи в цілому, виявити загрози і методи боротьби з ними, що найчастіше зустрічаються. Суттєва залежність сучасної діяльності фахівця у галузі інформаційної безпеки від надійного та ефективного контролю за даними визначає необхідність дослідження та вдосконалення існуючих засобів безпеки в базах даних SQL.

Будь-яку інформаційну систему можна як набір об'єктів, які взаємодіють із ядром всієї системи – інформацією. Кожен об'єкт може містити потенційну загрозу. Ефективно налаштована інформаційна система має враховувати всі джерела можливих загроз та вміти протистояти їм.

Визначаючи проблему, яка вирішується, важливо розглядати не лише технічний аспект, але й встановлювати зв'язок з об'єктом діяльності фахівця зі спеціальності. Аналіз стану проблеми дозволяє визначити рівень розв'язання задач, прогалини в наукових дослідженнях та невирішені завдання, що стають важливим фактором для обґрунтування мети та завдань кваліфікаційної роботи.

Ця кваліфікаційна робота присвячена розгляду безпеки інформаційної системи загалом та вузькому розгляду системи безпеки СУБД, як частини інформаційної системи.

Основні цілі даної роботи:

• дослідити загрози безпеки інформаційної системи загалом та визначити основні контрзаходи боротьби з загрозами у сучасних БД;

• дослідити особливості безпеки сучасних СУБД MySQL, Oracle, MS SQL Server;

• сформувати уявлення про розглянуті СУБД та здійснити порівняльний аналіз системи безпеки кожної з них та доцільність впровадження для посилення інформаційної безпеки.

Мета даного дослідження - систематизація та узагальнення інформації про існуючі засоби контролю інформаційної безпеки в базах даних SQL, а також розробка пропозицій щодо їх вдосконалення. Особлива увага приділяється теоретичній значущості отриманих результатів та їх прикладній цінності для індустрії, де використання СУБД здебільшого стало стандартом.

Ця робота має значення не лише у контексті поглибленого розуміння проблем безпеки в базах даних SQL, але і може слугувати важливим інструментом для розробників, адміністраторів та інших зацікавлених сторін, які працюють у галузі інформаційної безпеки.

РОЗДІЛ 1. СТАН ПИТАННЯ. ПОСТАНОВКА ЗАДАЧІ

1.1. Стан питання

Сучасні інформаційні системи, що використовуються на великих підприємствах, мають справу з гігабайтними обсягами даних (класичними прикладами інформаційних систем є банківські системи, системи резервування авіаційних або залізничних квитків, місць у готелях і т.д.). Це не дозволяє розміщувати їх у постійній і, тим більше, в оперативній пам'яті кожного комп'ютера, на якому вони потрібні. Крім того, ці дані є різнорідними, сильно пов'язані між собою та вимагають різноманітних способів свого вилучення та подання користувачеві. При роботі зі складно структурованими даними часто виникають проблеми їх дублювання та самоузгодженої зміни, а також низької швидкості доступу до даних. Подібні проблеми з великими труднощами вирішуються надбудовами над файловою системою, що вбудовуються в додатки. Щоб не повторювати одні й самі способи зберігання, вибору та модифікації складних даних у кожній бібліотеці мов програмування (чи, тим паче, у кожному додатку), виникли СУБД.

Можна вважати, що й прикладна інформаційна система спирається на деяку систему управління даними, вирішальну перелічені вище завдання, ця система управління даними є системою управління базами даних (СУБД).

СУБД – це програмне забезпечення, яке дозволяє кінцевим користувачам або розробникам додатків спільно використовувати дані та керувати ними. Вона представляє систематичний метод створення, оновлення, вилучення та зберігання інформації у базі даних. СУБД також, як правило, відповідає за цілісність, безпеку даних, контроль та оптимізацію доступу до даних, автоматичний відкат, перезапуск та відновлення.

Основні функції сучасних СУБД:

- Безпосереднє керування даними у зовнішній пам'яті. Ця функція включає забезпечення необхідних структур зовнішньої пам'яті як для зберігання даних, що безпосередньо входять до БД, так і для службових цілей, наприклад, для

прискорення доступу до даних у деяких випадках (зазвичай для цього використовуються індекси). У деяких реалізаціях СУБД активно використовуються можливості існуючих файлових систем, в інших роботах проводиться аж до рівня пристроїв зовнішньої пам'яті. Але підкреслимо, що у розвинених СУБД користувачі у разі не повинні знати, чи використовує СУБД файлову систему, і якщо використовує, те як організовані файли. Зокрема, СУБД підтримує власну систему найменування об'єктів БД.

- Управління буферами оперативної пам'яті. СУБД зазвичай працюють із базами даних значного розміру; принаймні, цей розмір зазвичай значно більший за доступний обсяг оперативної пам'яті. Зрозуміло, що якщо при зверненні до будь-якого елемента даних проводитиметься обмін із зовнішньою пам'яттю, то вся система працюватиме зі швидкістю пристрою зовнішньої пам'яті. Практично єдиним способом реального збільшення цієї швидкості є буферизація даних оперативної пам'яті. При цьому, навіть якщо операційна система виробляє загальносистемну буферизацію (як у випадку ОС UNIX), цього недостатньо для цілей СУБД, яка має набагато більшу інформацію про корисність буферизації тієї чи іншої частини БД. Тож у розвинених СУБД підтримується власний набір буферів оперативної пам'яті з дисципліною заміни буферів.

- Управління транзакціями. Транзакція (transaction) – це послідовність операцій над БД, що розглядаються СУБД як єдине ціле. СУБД підтримує паралельний доступом до даних, тобто. можливість одноразового звернення до однієї й тієї ж порції даних із боку кількох процесів. Щоб уникнути деяких небажаних наслідків такого звернення, СУБД реалізує механізми забезпечення ізольованості (isolation) транзакцій (транзакції виконуються незалежно одна від одної, ніби вони активізувалися суворо послідовно), їх атомарності (atomicity) (кожна транзакція або виконується повністю, або не виконується зовсім) та стійкості (durability) (системи містять засоби надійного збереження результатів виконання транзакцій та самовідновлення після різноманітних помилок та збоїв). Транзакції повністю реалізовані насамперед у СУБД, які розраховані на одночасну роботу багатьох користувачів (Oracle, MS SQL Server).

- Журналізація. Однією з основних вимог до СУБД є надійність зберігання даних у зовнішній пам'яті. Під надійністю зберігання розуміється те, що СУБД має бути в змозі відновити останній узгоджений стан БД після будь-якого апаратного чи програмного збою. Зазвичай розглядаються два можливі види апаратних збоїв: звані м'які збої, які можна трактувати як раптову зупинку роботи комп'ютера (наприклад, аварійне вимкнення живлення), і жорсткі збої, що характеризуються втратою інформації на носіях зовнішньої пам'яті. Прикладами програмних збоїв можуть бути: аварійне завершення роботи СУБД (внаслідок помилки в програмі або внаслідок деякого апаратного збою) або аварійне завершення програми користувача, внаслідок чого деяка транзакція залишається незавершеною. Першу ситуацію можна як особливий вид м'якого апаратного збою; у разі останньої потрібно ліквідувати наслідки лише однієї транзакції[9].

- Підтримка мов баз даних. Для роботи з базами даних використовуються спеціальні мови, в цілому звані мовами баз даних. У сучасних СУБД зазвичай підтримується єдина інтегрована мова, що містить всі необхідні засоби для роботи з БД, починаючи від її створення, і забезпечує базовий інтерфейс користувача з базами даних. Стандартною мовою найпоширеніших нині реляційних СУБД є мова SQL (Structured Query Language).

- Підтримка архітектур "клієнт-сервер", розподілених баз даних. Величезний пласт сучасних СУБД підтримує архітектуру клієнт-сервер, відповідно до якої запити, сформовані одним процесом (клієнтом), надсилаються для обробки іншого процесу (серверу). Найпростіший варіант даної архітектури передбачає, що СУБД цілком є сервер, крім інтерфейсів запитів, які взаємодіють з користувачем і відсилають запити та інші команди на сервер з метою виконання. Результати запитів повертаються клієнтам у вигляді таблиць чи відносин. Взаємозв'язок клієнта і сервера може бути набагато складнішим, особливо в тих випадках, коли результати виконання запитів мають складну структуру або великий обсяг. Концепція управління розподіленими базами даних полягає у забезпеченні засобу інтеграції локальних баз даних, що

знаходяться в деяких вузлах обчислювальної мережі, для того, щоб користувач, який працює в будь-якому вузлі мережі, мав доступ до всіх цих баз даних як до єдиної бази даних (приклади: банківська установа містить мережу відділень, спільноти університетів підтримують спільні цифрові бібліотеки, торгова компанія має єдину мережу магазинів).

- Можливість роботи багатьох користувачів та обмеження їх привілеїв. Діяльність сучасних СУБД пов'язана з оперуванням великими обсягами інформації, що провадиться широким колом осіб. З наявністю великої кількості користувачів різної кваліфікації пов'язана також підтримка цих СУБД привілеїв, які обмежують доступ до даних і цим підвищують надійність БД. Таким чином, забезпечується розмежування доступу суб'єктів з різними правами доступу до об'єктів різних рівнів конфіденційності.

Користувачів СУБД можна поділити на три групи:

- Адміністратори баз даних – утворюють особливу категорію користувачів СУБД. Вони створюють самі бази даних, здійснюють технічний контроль функціонування СУБД, забезпечують необхідну швидкодію системи. До обов'язків адміністратора, крім того, входить забезпечення користувачам доступу до необхідних їм даних, а також написання (або надання допомоги у визначенні) необхідних користувачеві зовнішніх уявлень даних. Адміністратор визначає правила безпеки та цілісності даних.

- Прикладні програмісти – відповідають за створення програм, які використовують базу даних. У сенсі захисту даних програміст може бути як користувачем, що має привілеї створення об'єктів даних та маніпулювання ними, так і користувачем, що має привілеї лише маніпулювання даними.

- Кінцеві користувачі бази даних – працюють із БД безпосередньо через термінал чи робочу станцію. Як правило, кінцеві користувачі мають обмежений набір привілеїв маніпулювання даними. Цей набір може визначатися при конфігуруванні кінцевого користувача інтерфейсу і не змінюватися. Політику безпеки в цьому випадку визначає адміністратор безпеки або адміністратор бази даних (якщо це одна і та ж посадова особа).

Серед програмних компонентів сучасної СУБД можна виділити внутрішню частину – ядро СУБД (Database Engine), компілятор мови БД, інтерпретатор та набір утиліт. Ядро СУБД зазвичай є резидентною програмою, що працює на сервері, надбудовою над його файловою системою. Ядро управляє буферами оперативної пам'яті, транзакціями та журналізацією; воно має власний інтерфейс, не доступним стороннім програмістам безпосередньо, але використовуваним в інших компонентах СУБД. Другий компонент – компілятор – транслює інструкції мови баз даних, але зазвичай над машинний код, що залежить від операційної системи, а платформно-незалежний код, виконуваний третім компонентом СУБД – інтерпретатором. Необхідність у четвертому компоненті – утилітах СУБД – виникає внаслідок низької швидкості виконання через мову баз даних деяких глобальних операцій над БД (наприклад, реплікації (імпорту/експорту) даних, резервного копіювання (backup), перевірки цілісності БД, перенесення її на нову версію СУБД) . Користувачі та адміністратори БД взаємодіють із СУБД через доступні інтерфейси[9].

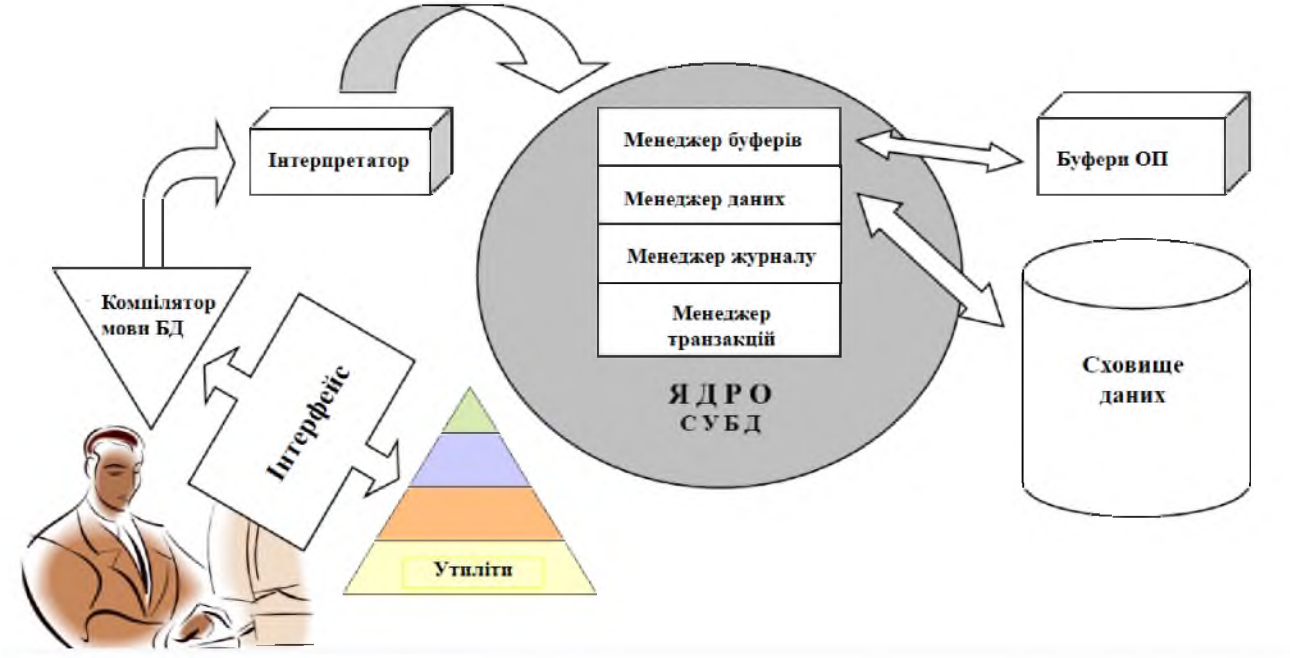

Рисунок 1.1 – Внутрішня структура сучасної СУБД

Системи управління базами даних є домінуючим інструментом для зберігання великих масивів інформації. Дані - цінний ресурс, доступ до якого

необхідно строго контролювати та регламентувати, так само як і до будь-яких інших ресурсів організації. Тому захист даних від несанкціонованого доступу є одним із пріоритетних завдань при проектуванні будь-якої інформаційної системи. Термін «захист» відноситься до захищеності баз даних від несанкціонованого доступу як навмисного, так і випадкового. Проте тема захисту баз даних включає як стандартні служби, що надаються цільової СУБД, а й значно ширше коло питань, які стосуються захищеності самих баз даних, і всього їх оточення.

Захист бази даних має на меті мінімізувати втрати, спричинені заздалегідь передбаченими подіями. Рішення, що приймаються, повинні забезпечити ефективне використання понесених витрат і виключати обмеження наданих користувачам можливостей.

Проломи в системі захисту можуть виникати в різних частинах інформаційної системи, що в свою чергу, наражає на небезпеку і власне базу даних. Отже, захист бази даних повинен охоплювати обладнання, програмне забезпечення, персонал і власне дані, що використовується. Для ефективної реалізації захисту необхідні відповідні засоби контролю, які визначаються конкретними вимогами, що випливають з особливостей системи, що експлуатується [15].

Усі можливі загрози безпеці бази даних можна розглянути з погляду результатів, яких вони можуть привести:

- викрадення та фальсифікація даних
- Втрата конфіденційності (порушення таємниці)
- Порушення недоторканності особистих даних
- Втрата цілісності
- Втрата доступності

Зазначені ситуації відзначають основні напрями, у яких повинні вживатися заходи, що знижують ступінь ризику. У деяких ситуаціях всі зазначені аспекти пошкодження даних тісно пов'язані між собою, так що дії, спрямовані на порушення захищеності системи в одному напрямку, часто призводять до

зниження її захищеності у всіх інших. Крім того, деякі події - наприклад, порушення недоторканності особистих даних або фальсифікація - можуть виникнути внаслідок як навмисних, так і ненавмисних дій і зовсім необов'язково супроводжуються будь-якими змінами в базі даних або системі, які можна буде виявити тим чи іншим способом.

Викрадення та фальсифікація даних можуть відбуватися не тільки в середовищі бази даних – вся інформаційна система так чи інакше схильна до цього ризику. Однак дії з викрадення або фальсифікації інформації завжди здійснюються людьми, тому основна увага має бути зосереджена на скороченні зручних ситуацій для виконання таких дій.

Втрата конфіденційності та порушення недоторканності особистих данних. Поняття конфіденційності означає необхідність збереження даних у таємниці. Як правило, конфіденційними вважаються ті дані, які є критичними для всієї організації, тоді як поняття недоторканності даних стосується вимог захисту інформації про окремих її працівників. Наслідком порушення в системі захисту, що викликало втрату конфіденційності даних, може бути втрата позицій у конкурентній боротьбі, тоді як наслідком порушення недоторканності особистих даних будуть юридичні заходи щодо організації.

Втрата цілісності даних призводить до спотворення чи руйнації даних, що може мати найсерйозніші наслідки подальшої роботи організації. В даний час безліч організацій функціонує в безперервному режимі, надаючи послуги клієнтам по 24 години на добу та по 7 днів на тиждень. Втрата доступності даних означатиме, що або дані, або система, або й те, й інше одночасно виявляться недоступними користувачам, що може наражати на небезпеку саме подальше існування організації. У деяких випадках ті події, які спричинили переход системи в недоступний стан, можуть одночасно викликати і руйнування даних у базі даних.

Загроза може бути викликана ситуацією або подією, здатною завдати шкоди організації, причиною якої може бути людина, подія чи збіг обставин. Шкода може бути очевидною (наприклад, втрата обладнання, програмного

забезпечення або даних) або неочевидною (наприклад, втрата довіри партнерів чи клієнтів). Перед кожною організацією стоїть проблема з'ясування всіх можливих небезпек, що у деяких випадках зовсім непросте завдання. Тому виявлення хоча б найважливіших загроз може знадобитися досить багато часу і зусиль, що слід враховувати.

Будь-яка небезпека має розглядатися як потенційна можливість порушення системи захисту, яка у разі своєї реалізації може мати той чи інший негативний вплив. Хоча деякі типи небезпек можуть бути як навмисними, так і ненавмисними, результати в будь-якому випадку будуть однакові - вони призводитимуть до втрати таємності, або до порушення цілісності, або до втрати доступності.

Організація повинна встановити типи можливих небезпек, яким може наразитися її комп'ютерна система, після чого розробити відповідні плани та необхідні заходи, з оцінкою рівня витрат, необхідних для їх реалізації. Безумовно, витрати часу, зусилля та грошей навряд чи виявляться ефективними, якщо вони стосуватимуться потенційних небезпек, здатних завдати організації лише незначних збитків. Ділові процеси організації можуть бути схильні до таких небезпек, які неодмінно слід враховувати, однак, частина з них може мати місце у рідкісних ситуаціях. Тим не менш, навіть такі малоймовірні обставини повинні бути взяті до уваги, особливо якщо їх вплив може виявитися дуже суттєвим. На малюнку 2 складено узагальнену схему потенційних небезпек, яким може наражатися типова комп'ютерна система.

Існуючі приклади проломів, що виявилися в захисті комп'ютерних систем, демонструють той факт, що навіть найвищого рівня захищеності самої системи може виявитися недостатньо, якщо все ділове середовище не матиме необхідного рівня захисту. Метою є досягнення балансу між обґрунтованим рівнем реалізації захисних механізмів, функціонування яких не викликає зайвих обмежень у роботі користувачів та витратами на їх підтримку.

Найчастіше саме випадкові небезпеки є причиною основної частини втрат у більшості організацій. Будь-який випадковий інцидент, що спричинив

порушення системи захисту, повинен бути зафіксований у документації, із зазначенням відомостей про персони, пов'язані з його появою. Іноді подібні записи повинні аналізуватися з метою встановлення частоти виникнення подібних інцидентів, пов'язаних з тими самими людьми. Отримані відомості слід використовувати для уточнення існуючих процедур та встановлених обмежень. Наприклад, роз'єднання або обрив кабелів, що повторюється, повинен послужити причиною перегляду способу укладання або маршруту їх проведення. Аналогічно, повторюване внесення до системи вірусів та іншого небажаного програмного забезпечення має бути розцінене як серйозна загроза, однак тут можуть мати місце труднощі з оцінкою випадковості або навмисності їх появи. Проте, можуть бути розроблені процедури, призначені для перевірки всього нового програмного забезпечення та носіїв, що надходять у розпорядження організації. Вони можуть бути доповнені обмеженнями, які забороняють працівникам використовувати власне програмне забезпечення. Після вжиття подібних заходів появу в системі вірусів навряд чи можна буде розцінювати як випадкову.

Типи можливих загроз лежать у широкому діапазоні — від пожеж і повеней, що безпосередньо впливають на всі елементи системи, до інцидентів, що впливають лише на її частину. Наприклад, випадкова відмова роботи однієї робочої станції може блокувати роботу лише одного користувача і не вплинути на стан програмного забезпечення або даних.

У контексті розробки засобів контролю інформаційної безпеки важливим є також аналіз специфічних внутрішніх загроз, які можуть виникати внаслідок дій або бездіяльності співробітників. Це може включати ненавмисне поширення конфіденційної інформації, неправильне використання корпоративних ресурсів, або випадкове внесення помилок в бази даних. Розробка ефективних процедур та протоколів для моніторингу та аудиту діяльності співробітників є ключовим елементом забезпечення безпеки інформаційних систем.

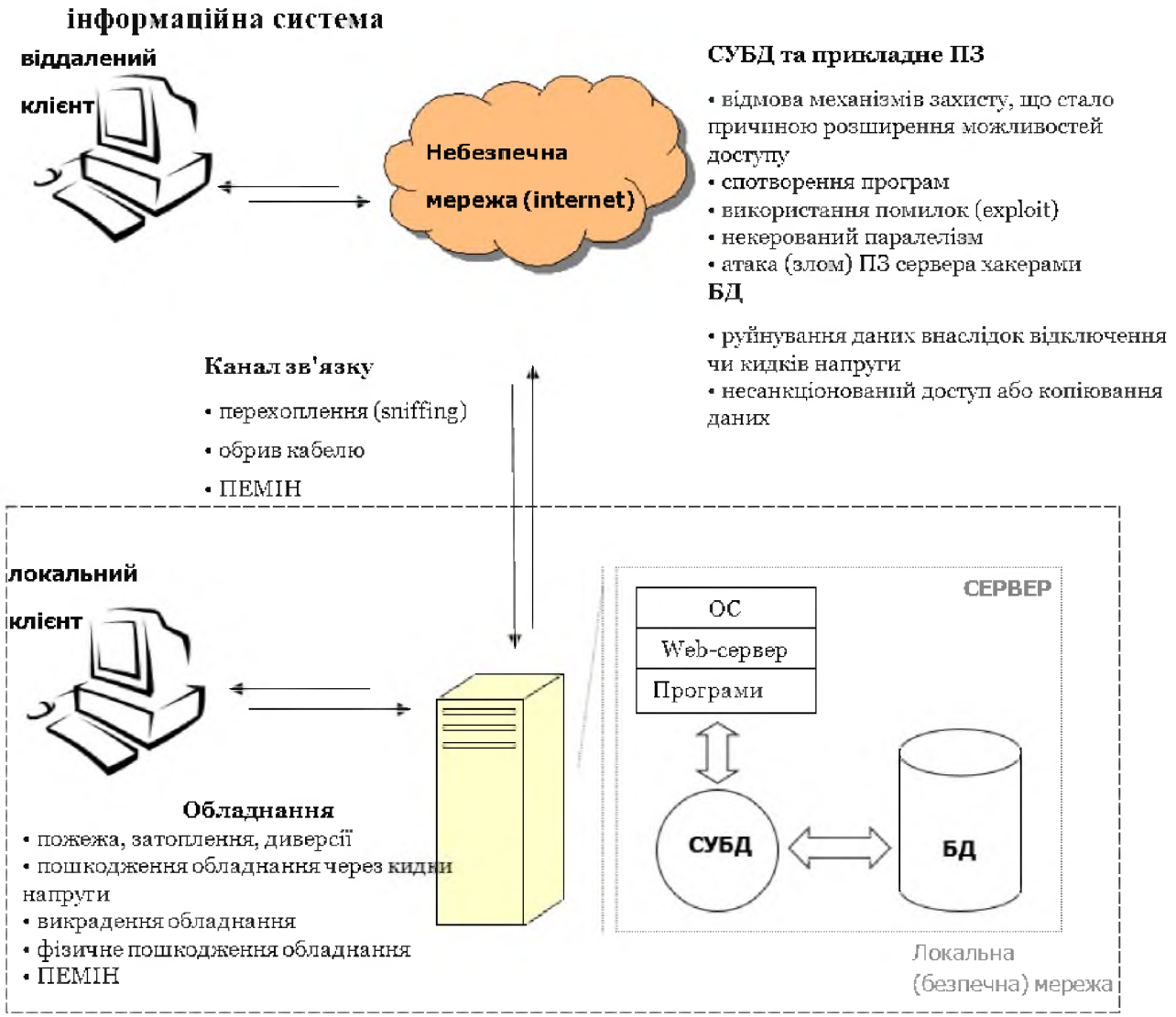

Рисунок 1.2 – Узагальнена схема потенційних загроз, яким схильна інформаційна система

Також серед тих кто взаємодіє з базою даних можна виділити 3 категорії співробітників: користувачі, програмісти та адміністратори бази даних.

Кожна з цих категорій відіграє унікальну роль у забезпеченні інформаційної безпеки. Користувачі взаємодіють із даними на повсякденній основі, тоді як програмісти відповідають за розробку та підтримку програмного забезпечення. Адміністратори баз даних, у свою чергу, забезпечують управління,

налаштування та оптимізацію систем, граючи ключову роль у підтримці безперебійної та безпечної роботи баз даних.

Користувачі

- використання чужих логінів
- недостатня професійна підготовка
- нечесні/ображені співробітники
- використання вірусів

Програмісти/Оператори БД

- створення лазівок (trapdoor)
- спотворення програм (наприклад, вимкнення засобів захисту)
- недостатня професійна підготовка
- недостатній рівень обмежень та процедур захисту

Адміністратори

- неадекватна політика безпеки та процедур обслуговування СУБД

Рисунок 1.3 – Ризики для інформаційної системи поділені на категорії тих, кто взаємодіє з базою даних

Користувачі, програмісти та адміністратори баз даних SQL представляють потенційні загрози для інформаційної безпеки. У разі користувачів виникає ризик несанкціонованого доступу та введення шкідливих даних, що може призвести до порушення конфіденційності інформації. Програмісти, в свою чергу, можуть створювати уразливості через недостатню перевірку вводу або витоки коду, спричиняючи можливість SQL-ін'єкцій та інших атак.

Адміністратори баз даних мають ключову роль у забезпеченні безпеки, але вони також можуть бути джерелом загроз, особливо при наданні несанкціонованого доступу чи невиправлених уразливостей. Невірно налаштовані адміністраторські права можуть відкрити доступ до конфіденційної інформації.

Загальна усвідомленість загроз з боку цих трьох груп та вжиття заходів

безпеки, таких як регулярне оновлення та моніторинг системи, є ключовим для ефективного управління ризиками та забезпечення безпеки баз даних SQL.

Загалом можна звести небезпеки у одну зведену таблицю.

Таблиця 1.1. Зведена таблиця небезпек із зазначенням результатів їхньої дії та методів боротьби з ними

Небезпека	Результат	Методи боротьби
1	2	3
Устаткування		
• пожежа, затоплення, диверсії	ЦД	Грамотна система фізичної безпеки підприємства (протипожежні датчики, системи фізичної охорони та спостереження)
• пошкодження обладнання через кидки напруги	ЦД	Стабілізатори напруги, джерела безперебійного живлення (UPS)
• викрадення обладнання	СД	Системи фізичної охорони та спостереження
• фізичне пошкодження обладнання	ЦД	Аккуратность в обслуживании, резервные копии
• Побічні електромагнітні випромінювання та наведення	С	Шифрування та використання екранування приміщення (обладнання)
Персонал		
• недостатня професійна підготовка	С	Курси + екзамени для персоналу перед допуском до роботи з інструментами СУБД
• нечесні/ображені співробітники	СЦД	Аудит, грамотний контроль доступу
• Використання вірусів	ЦД	Використання антивірусних систем захисту з постійним автоматичним оновленням антивірусних баз
• створення лазівок (trapdoor)	С	Аудит, грамотний контроль доступу
• спотворення програм (наприклад,	С	Аудит

вимкнення засобів захисту)		
• недостатній рівень обмежень та процедур захисту	СЦД	Аудит, ретельне тестування настроєної системи захисту перед запуском її в експлуатацію
Канал зв'язку		
• перехоплення (sniffing)	С	Шифрування інформації, що передається, використання захищеного каналу зв'язку
• обрив кабелю	Д	Наявність запасної лінії передачі
• Побічні електромагнітні випромінювання та наведення	С	Шифрування та використання екранованих ліній передач
СУБД та прикладне ПЗ		
• Відмова механізмів захисту, що стало причиною розширення можливостей доступу	С	Детальний план поновлення, уважне читання документації до оновлення, тестування, аудит, запущеної в робочий стан системи
• спотворення програм	С	Аудит
• exploit' и використання помилок ПЗ)	СЦД	Своєчасне оновлення ПЗ, Аудит, ретельне тестування прикладних програм перед запуском їх у роботу з СУБД
• некерований паралелізм	Ц	Використання транзакцій
• атака(злом) ПЗ сервера хакерами	СЦД	Використання брендмауерів (firewall) на стороні сервера, використання процедур, що зберігаються
БД		
• руйнування даних внаслідок відключення або кидків напруги	ЦД	Створення резервних копій, використання UPS
• несанкціонований доступ або копіювання даних	С	Контроль фізичного доступу до баз даних, автентифікація

С – втрата секретності (викрадення, фальсифікація даних, втрата конфіденційності); Ц – порушення цілісності інформації, Д – відмова доступності даних

1.2. Постановка задач

Сучасні сховища даних компаній зазвичай складаються з двох ключових компонентів: самої бази даних, де зберігається інформація, та програмного забезпечення для її керування та адміністрування. Забезпечення безпеки цих систем є критично важливим, і це вимагає створення умов для безпечного управління збереженими даними. Питання захисту сховищ даних можна розділити на дві категорії: ті, що не залежать від даних, та ті, що залежать.

Вразливості, не залежні від даних, схожі на ті, що зустрічаються у багатьох інших програмних продуктах. Це може бути пов'язано з різними причинами, такими як нерегулярне оновлення програмного забезпечення, недостатній досвід або навички системного адміністратора, або наявність невикористовуваних функцій.

З іншого боку, більшість питань безпеки сховищ даних тісно пов'язані з самими даними. Наприклад, деякі системи управління базами даних (СУБД) дозволяють писати запити за допомогою спеціальних мов, які мають власні набори методів для взаємодії з даними. Архітектура цих мов пов'язана з моделлю даних, використаною для зберігання інформації. Таким чином, модель даних частково визначає особливості мови запитів, яка, у свою чергу, може впливати на наявність певних вразливостей. Види вразливостей, такі як ін'єкції SQL або Java, використовуються різними способами в залежності від особливостей синтаксису мови.

У сфері безпеки даних з'являються та існують численні та різноманітні засоби забезпечення безпеки, що обумовлено різними еволюційними особливостями. Ця різноманітність призвела до відсутності єдиного, комплексного підходу до розуміння безпеки даних. Такий стан речей ускладнює не тільки визначення загальних принципів та методів захисту сховищ даних, але

й прогнозування майбутніх атак та розробку відповідних механізмів захисту.

Однією з ключових проблем є те, що, незважаючи на постійний розвиток нових технологій та методів захисту, багато сучасних систем все ще вразливі до давно відомих типів атак. Це ставить під сумнів ефективність існуючих систем захисту та вимагає постійного вдосконалення та оновлення безпекових механізмів.

Крім того, існує проблема підготовки фахівців у галузі безпеки. Через швидкий розвиток технологій та постійну зміну загроз, спеціалістам у сфері безпеки потрібно постійно оновлювати свої знання та навички. Це включає в себе не тільки розуміння новітніх технологій, але й уміння адаптуватися до нових типів загроз та ефективно використовувати інноваційні методи захисту.

Багато вразливостей у системах баз даних залишаються актуальними через недостатню увагу або незнання питань безпеки з боку адміністраторів цих систем. Одним з яскравих прикладів є прості SQL-ін'єкції, які активно експлуатуються у вразливих веб-додатках, де не приділяється достатньо уваги фільтрації та перевірці вхідних даних.

Застосування різних засобів інформаційної безпеки становить для організацій фінансовий компроміс. Впровадження більш захищених продуктів та найм більш кваліфікованого персоналу вимагає значних інвестицій. Окрім цього, компоненти безпеки часто можуть негативно впливати на продуктивність систем управління базами даних (СУБД). Це означає, що забезпечення високого рівня безпеки може призводити до зниження швидкості та ефективності роботи баз даних.

Тому організаціям потрібно знаходити баланс між забезпеченням безпеки та підтриманням продуктивності систем. Важливо також регулярно проводити навчання та підвищення кваліфікації персоналу, оскільки багато вразливостей можна усунути, належно керуючи системами та правильно налаштовуючи безпекові механізми.

Для ефективного вирішення проблем інформаційної безпеки в системах управління базами даних (СУБД) необхідно перейти від простого закриття

окремих вразливостей до більш комплексного підходу у забезпеченні безпеки сховищ інформації. Комплексний підхід включає створення і застосування універсальних методик, які можна використовувати на різних етапах розробки та впровадження сховищ даних і користувацького програмного забезпечення.

Застосування таких методик дозволить уникнути багатьох помилок у процесі управління СУБД та забезпечити захист від найбільш розповсюджених вразливостей. Окрім того, класифікація загроз і вразливостей допоможе систематизувати їх для подальшого аналізу та розробки стратегій захисту. Це надасть фахівцям з безпеки можливість зрозуміти причинно-наслідкові зв'язки між різними вразливостями і причинами їх виникнення.

Впровадження конкретних механізмів захисту в СУБД дозволить адміністраторам і розробникам краще прогнозувати потенційні загрози, пов'язані з цими механізмами, та заздалегідь розробляти відповідні стратегії захисту. Такий підхід вимагає більш глибокого розуміння безпеки на всіх рівнях організації, а також включає не тільки технічні, але й організаційні заходи, що сприяють підвищенню загального рівня безпеки інформаційних систем.

В області захисту баз даних існує цілий спектр технологій і методів, які забезпечують захист від різних видів атак. Ефективність цих методів залежить від багатьох факторів, включаючи конфігурацію бази даних і сервера, на якому вона розміщена, правильне проектування та реалізація IT-інфраструктури, топологію мережі та людський фактор, включаючи лояльність персоналу.

Засоби захисту баз даних поділяються на основні та додаткові.

Основні засоби захисту включають:

- Парольний захист: Забезпечує базовий рівень безпеки, обмежуючи доступ до бази даних за допомогою паролів.
- Захист полів і записів таблиць БД: Включає обмеження доступу до певних частин бази даних.
- Встановлення прав доступу до об’єктів БД: Дозволяє контролювати, хто має доступ до певних даних або функцій бази даних.
- Шифрування даних і програм: Перешкоджає несанкціонованому доступу

до даних, навіть якщо зловмисникам вдається отримати доступ до фізичної бази даних.

Додаткові засоби захисту включають:

- Вбудовані засоби контролю значень даних: Перевіряють правильність та актуальність даних відповідно до їх типів.
- Підвищення достовірності вводимих даних: Забезпечує, що дані, що вводяться в систему, є точними і надійними.
- Забезпечення цілісності зв’язків таблиць: Важливо для підтримки правильних і послідовних зв'язків між різними частинами бази даних.
- Організація спільного використання об’єктів БД у мережі: Важливо для розподілених систем, де кілька користувачів або групи користувачів мають доступ до однієї бази даних.

Важливо зауважити, що ефективний захист баз даних вимагає комплексного підходу, що включає як технічні, так і організаційні заходи. Важливість такого підходу зростає у відповідь на постійно змінні та ускладнені загрози у сучасному цифровому світі.

Експерти компанії Application Security вказують на 10 основних загроз безпеці баз даних, які часто ігноруються IT-персоналом. Ці загрози включають:

- Використання стандартних, порожніх або слабких паролів і логінів: Це створює легку мішень для зловмисників, які можуть використовувати розповсюджені або легко вгадувані паролі для доступу до систем.
- SQL-ін’єкції: Цей тип атаки дозволяє зловмисникам вводити або "ін'єктувати" шкідливий SQL-код у запит, що може призвести до неправомірного доступу або маніпуляцій з даними.
- Розширені користувацькі і групові права: Надмірні права доступу можуть дозволити користувачам або групам виконувати операції, які повинні бути обмежені.
- Активізація невикористовуваних функцій БД: Наявність активних, але невикористовуваних функцій може стати потенційним входом для атак.

- Порушення в управлінні конфігураціями: Неналежне управління конфігураціями може залишити систему вразливою до атак.
- Переповнення буфера: Цей тип вразливості може дозволити зловмисникам виконувати довільний код або зупинити систему.
- Ескалація привілеїв: Атака, що дозволяє зловмиснику з низьким рівнем доступу отримати вищі привілеї.
- DoS-атаки (Denial of Service): Ці атаки заблоковують ресурси системи, роблячи їх недоступними для легітимних користувачів.
- Несвоєчасне оновлення ПЗ: Відсутність своєчасних оновлень може залишити систему вразливою до відомих вразливостей.
- Відмова від шифрування даних на стаціонарних і мобільних пристроях: Незашифровані дані можуть бути легко перехоплені або скомпрометовані під час передачі або у разі втрати пристрою.

Ці загрози підкреслюють важливість комплексного підходу до безпеки, що включає як технічні, так і організаційні заходи, а також регулярне навчання персоналу і оновлення політик безпеки.

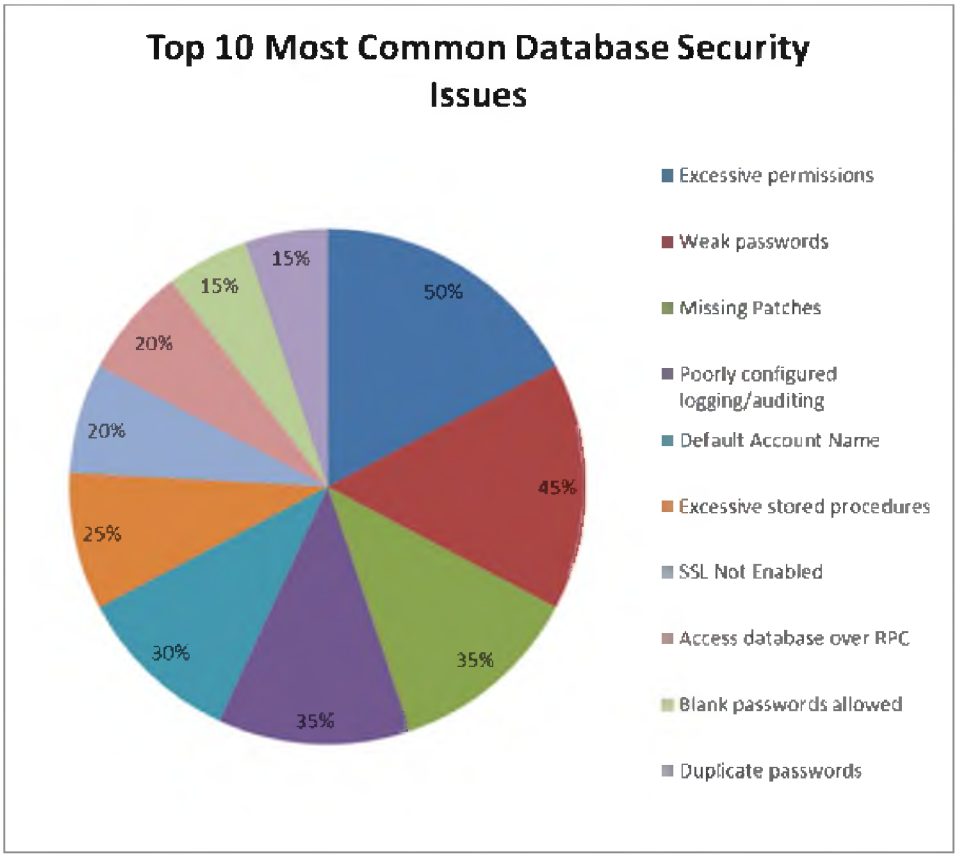

Рисунок 1.4 – 10 найбільш поширених вразливостей у БД

Цей перелік відображає різноманітність і складність проблем, з якими стикаються IT-спеціалісти при управлінні безпекою баз даних. Розглянемо їх детальніше:

1. Надмірні привілеї: Надання користувачам ширших привілеїв, ніж необхідно, може призвести до зловживань або несанкціонованого доступу до чутливих даних.
2. Слабкі паролі: Використання простих або стандартних паролів робить бази даних вразливими до атак, таких як брутфорс або вгадування пароля.
3. Відсутні патчі: Невстановлення оновлень безпеки або працювання на застарілих версіях ПЗ збільшує ризик використання відомих вразливостей зловмисниками.
4. Погано налаштований аудит: Неналежне ведення аудиту може ускладнити виявлення та реагування на інциденти безпеки.
5. Назва облікового запису за замовчуванням: Невикористання або неправильне управління обліковими записами за замовчуванням може створити легкі цілі для атак.
6. Надмірна кількість збережених процедур: Використання зайвих збережених процедур, особливо тих, що можуть виконувати системні команди або доступ до файлів ОС, створює додаткові ризики.
7. Не ввімкнений SSL: Передача даних у незашифрованому форматі може призвести до їх перехоплення або маніпуляції.
8. Дозволені порожні паролі: Навіть можливість використання порожніх паролів є значним ризиком безпеки.
9. Дублікати паролів: Використання однакових або подібних паролів для різних облікових записів знижує загальний рівень безпеки.
10. Первинний номер облікового запису є звичайним текстом: Зберігання важливих облікових даних у незашифрованому форматі робить їх легкодоступними для несанкціонованого доступу.

Кожна з цих проблем вимагає відповідального підходу та належного управління, що включає як технічні, так і організаційні заходи безпеки. Це підкреслює важливість комплексного підходу до безпеки баз даних.

У сучасному світі, де обсяги даних зростають з кожним днем, їхній захист стає критично важливим завданням. Бази даних SQL є одним із основних інструментів для зберігання та обробки інформації в різних сферах, від бізнесу до державного управління. Однак збільшення залежності від електронних даних супроводжується зростанням загроз їхній безпеці. Порушення безпеки можуть призвести до значних фінансових втрат, втрати конфіденційності та навіть загрози національній безпеці.

У світлі цих викликів дослідження засобів контролю інформаційної безпеки в базах даних SQL набуває особливої актуальності. Це дослідження спрямоване на аналіз існуючих методів захисту даних, оцінку їх ефективності та розроблення рекомендацій для підвищення рівня безпеки інформаційних систем. Особливу увагу приділено аспектам шифрування даних, управління доступом та моніторингу активності у базах даних.

Основною метою цього дослідження є розробка комплексного підходу до підвищення безпеки в SQL базах даних. Очікується, що результати роботи допоможуть краще зрозуміти поточні загрози та вразливість у системах баз даних, а також запропонують ефективні способи їхньої нейтралізації. Конкретні очікувані результати включають:

- Огляд сучасних методів забезпечення безпеки: Подання аналізу поточних технологій та практик у галузі безпеки баз даних.
- Аналіз уразливостей: Виявлення найчастіших і найкритичніших уразливостей у SQL-базах даних, виходячи з недавніх інцидентів та досліджень.
- Рекомендації щодо покращення безпеки: Розробка конкретних рекомендацій щодо посилення захисту даних, включаючи методи шифрування, політики доступу та моніторингу.
- Напрямки для подальших досліджень: Визначення потенційних напрямків для майбутніх досліджень у сфері інформаційної безпеки баз даних.

Це дослідження прагне зробити внесок у підвищення загального рівня інформаційної безпеки, що є актуальним як для фахівців у галузі IT, так і для організацій, які використовують SQL-бази даних у своїй діяльності.

1.3. Висновки

У даному розділі було проведено ретельний огляд існуючих досліджень та практик у галузі інформаційної безпеки SQL-баз даних. Цей аналіз виявив, що, незважаючи на значні зусилля та досягнення у забезпеченні безпеки даних, як і раніше, існують значні вразливості та виклики. Ці уразливості варіюються від технічних недоліків до складних стратегій кібератак, потребують постійного відновлення захисних заходів. Також були чітко визначені цілі та завдання дослідження. Головною метою є розробка та пропозиція покращених методів та стратегій для підвищення рівня інформаційної безпеки в SQL-базах даних. Через аналіз існуючих уразливостей, вивчення передових практик і розробку нових підходів, це дослідження прагне зробити значний внесок у сферу інформаційної безпеки баз даних.

Суть цих задач наголошує на актуальності та необхідності глибокого дослідження в даній галузі, а також задає чіткі рамки для подальшого аналізу та розробки рекомендацій. З виявлених уразливостей і поточних тенденцій у сфері інформаційної безпеки, це дослідження спрямоване створення ефективних і практично застосовних рішень захисту даних у SQL-базах.

РОЗДІЛ 2. СПЕЦІАЛЬНА ЧАСТИНА

2.1. Некомп’ютерні засоби контролю

Некомп'ютерні засоби контролю - це методи, не пов'язані з комп'ютерною підтримкою і спрямовані на вироблення обмежень, угод та інших адміністративних заходів, вжитих в організації. Вони включають:

• заходи щодо безпеки;

• планування захисту від непередбачених обставин;

• контроль за персоналом;

• захист приміщень та сховищ;

• Контроль за фізичним доступом.

Заходи забезпечення безпеки передбачають вичерпне визначення коштів, з яких забезпечуватиметься захист обчислювальної системи цієї організації. Кожна організація має підготувати та реалізувати документальний перелік заходів забезпечення безпеки.

У такому документі має бути визначено таке:

- область ділових процесів організації, на яку вони встановлюються;
- відповідальність та обов'язки окремих працівників;
- дисциплінарні заходи, які вживаються у разі виявлення порушення встановлених обмежень;
- процедури, які мають обов'язково виконуватися.

Процедури, які визначаються як заходи безпеки, можуть вимагати розробки інших процедур, визначення яких виходить за рамки цього документа. У такому разі документ з визначенням заходів безпеки постійно зберігає свою значущість, хоча може бути потрібний його періодичний перегляд, тоді як виконувані процедури реалізації цих вимог повинні модифікуватися при кожному внесенні змін до системи або модернізації використовуваної технології.

Планування захисту від непередбачених обставин розробляється з метою докладного визначення послідовності дій, необхідних виходу з різних незвичайних ситуацій, не передбачених процедурами нормального

функціонування системи – наприклад, у разі пожежі чи диверсії. У системі може існувати єдиний план захисту від непередбачених обставин, а може бути кілька – кожен окремо. Типовий план захисту від непередбачених обставин повинен включати такі елементи, як:

• відомості про те, хто є головною відповідальною особою та як можна встановити з нею контакт;

• інформація про те, хто і на якій підставі ухвалює рішення про виникнення незвичайної ситуації;

• технічні вимоги до передачі управління резервним службам, які можуть включати:

• відомості про розташування альтернативних сайтів;

• відомості про необхідне додаткове обладнання;

• інформацію про необхідність додаткових ліній зв'язку.

• організаційні вимоги щодо персоналу, який здійснює передачу управління резервним службам;

• відомості про будь-які зовнішні джерела, в яких можна буде отримати допомогу, наприклад, про постачальників обладнання;

• відомості про наявність страховки у разі даної конкретної ситуації.

Будь-який розроблений план захисту від непередбачених обставин повинен періодично переглядатися та тестуватися щодо його здійсненності.

Творці комерційних СУБД покладають всю відповідальність за ефективність управління системою її користувачів. Тому, з погляду захисту системи, виключно важливу роль відіграють ставлення до справи та дії людей, які безпосередньо залучені до цих процесів. Важливість цього зауваження підкреслюється тим, що основний ризик у будь-якій організації пов'язані з діями, виробленими співробітниками самої організації, а чи не з можливими зовнішніми загрозами, слід, що забезпечення необхідного рівня контролю над персоналом дозволяє мінімізувати можливий ризик.

Захист приміщень та сховищ

Основне обладнання системи, включаючи принтери, якщо вони

використовуються для друку конфіденційної інформації, повинно розміщуватися в приміщенні з обмеженим доступом, який повинен бути дозволений тільки основний фахівцям. Решта обладнання, особливо переносне, має бути надійно закріплене в місці розміщення та забезпечене сигналізацією. Однак умова тримати приміщення постійно замкненим може виявитися небажаною або неможливою, оскільки працівникам може бути потрібний постійний доступ до встановленого там обладнання.

Для будь-якої організації життєво необхідно мати надійно захищене місце, де зберігатимуться копії програм, резервні копії системи, інші архівні матеріали та документація. Переважно, щоб це приміщення знаходилося на майданчику, відмінному від місця розташування основного обладнання системи. Усі носії інформації, включаючи диски і магнітні стрічки, повинні зберігатися в сейфах, що не згоряються. Аналогічно повинна зберігатися і документація. Все, що зберігається в такому приміщенні, має бути зареєстроване у спеціальному каталозі, із зазначенням дати приміщення носія або документа на зберігання. Періодичність, з якою нові матеріали будуть розміщені в подібному архіві, повинна регламентуватися розробленими процедурами копіювання. Крім того, організації можуть використовувати і зовнішні сховища, періодично доставляючи туди новостворені резервні копії та інші архівні матеріали, причому частота доставки також повинна визначатися встановленими нормами та процедурами.

Контроль за фізичним доступом призначений для управління тим, хто матиме доступ до певних приміщень, до певного обладнання чи будівлі організації. Наприклад, найбільш "відповідальні" приміщення (скажімо, ті, в яких встановлені основні комп'ютери) можуть бути обладнані системами вхідного контролю. У подібних системах можуть використовуватися різні принципи — наприклад, спеціальні ключі, картки, кодові замки або засоби вказівки пароля. Найбільш складні комплекси передбачають використання відбитків пальців, знімків райдужної оболонки очей, фіксацію особливостей голосу чи почерку. Також можуть бути вжиті заходи щодо контролю доступу

поза будівлями для обмеження доступу в окремі будівлі. Для спостереження за територією може використовуватись спеціальна охорона, в обов'язки якої входить контроль входу/виходу персоналу та відвідувачів організації. Слід зазначити, що основним завданням будь-яких механізмів контролю за фізичним доступом є забезпечення їх ефективності, проте необхідна ефективність має досягатися без створення додаткових перешкод персоналу у виконанні їхніх службових обов'язків. В іншому випадку зростає небезпека пошуку персоналом всіляких шляхів обходу встановлених вимог.

На закінчення розділу хочеться додати, що з проектуванні та реалізації політики безпеки слід враховувати[4], що з комерційних організацій потенційні загрози порядку зменшення розмірів збитків розташовуються так:

1. Помилки та упущення обслуговуючого персоналу та користувачів.
2. Дії нечесних працівників.
3. Вогонь.
4. Дії скривджених працівників.
5. Вода.
6. Дії сторонніх осіб.

Основну увагу слід звертати на систематизацію та автоматизацію дій адміністраторів баз даних та сервера СУБД. Великий відсоток ручної роботи неминуче призведе до помилок (ненавмисних), які можуть виявитися страшнішими за пожежу.

Захист від нечесних та скривджених працівників вимагає, насамперед, грамотної постановки служби обліку та реагування на підозрілі ситуації. Важлива, зрозуміло, і психологічна атмосфера, що панує у колективі.

Хочеться ще раз наголосити на необхідності систематичного підходу до забезпечення інформаційної безпеки настільки складного об'єкта, яким є сучасний сервер баз даних.

2.2. Комп’ютерні засоби контролю

Незважаючи на широкий діапазон комп'ютерних засобів контролю,

доступних в даний час на ринку, загальний рівень захищеності СУБД визначається можливостями операційної системи, що використовується, оскільки робота цих двох компонентів тісно пов'язана між собою. Сувора автентифікація є наріжним каменем будь-якого плану реалізації безпеки. Неможливо керувати авторизацією та відстежувати використання ресурсів без цього.

Аутентифікація

Аутентифікація – механізм визначення того, чи користувач тим, за кого себе видає.

За надання користувачам доступу до комп'ютерної системи зазвичай відповідає системний адміністратор, обов'язки якого входить створення облікових записів користувачів. Кожному користувачеві надається унікальний ідентифікатор (ID), який використовується операційною системою для визначення того, хто є хто. З кожним ідентифікатором пов'язується певний пароль, який вибирає користувач і невідомий операційній системі. При реєстрації користувач повинен надати системі свій пароль для перевірки (автентифікації) того, чи є він тим, за кого себе видає.

Подібна процедура дозволяє організувати контрольований доступ до комп'ютерної системи, але не обов'язково надає право доступу до СУБД або іншої прикладної програми. Для отримання користувачем права доступу до СУБД може використовуватись окрема подібна процедура. Відповідальність за надання прав доступу до СУБД зазвичай несе адміністратор бази даних, обов'язок якого входить створення індивідуальних ідентифікаторів користувачів, але цього разу вже в середовищі самої СУБД. Кожен із ідентифікаторів користувачів СУБД також пов'язується з паролем, який має бути відомий лише даному користувачеві. Цей пароль використовуватиметься підпрограмами СУБД для ідентифікації користувача.

Використання паролів є найбільш поширеним методом аутентифікації користувачів, тому одним із способів забезпечення правильного використання СУБД є розробка та розповсюдження порад щодо створення правильних паролів:

Уникати паролів, які надто прості. Кожен пароль повинен складатися принаймні з шести символів, а якщо можливо, то й більше.

Кожен пароль повинен складатися принаймні з комбінації символів алфавіту та цифрових знаків. Використання інших допустимих символів робить розгадування пароля ще складнішим.

• Уникати створення пароля, який є цілим словом (як рідною мовою користувача, так і будь-якою іноземною мовою).

• Не вставляйте особисту статистику у пароль. Адреси, номери соціального страхування, телефонні номери тощо легко можуть бути розгадані та не підходять для паролів.

• Подумати про поєднання двох незв'язаних слів із символом чи цифрою між ними. Наприклад, пароль "toeSstor" є відповідним паролем.

• Використовувати менімонічні прийоми, які допоможуть запам'ятати пароль. Наприклад, можна використовувати таку пропозицію, як "Moondance Вана Моррісона - мій улюблений альбом", щоб запам'ятати, що пароль - "mвммла" (перші літери кожного слова в реченні).

• Час від часу змінюються паролі.

Коли користувачеві СУБД доступ до СУБД більше не потрібний або якщо він йде з компанії, адміністратор баз даних повинен видалити його обліковий запис із системи якнайшвидше. Проте це може стати складним завданням - обліковий запис не може бути видалений, якщо людина в даний час використовує базу даних або якщо користувач є власником будь-якого з об'єктів бази даних.

З цієї причини мудро обмежити користувачів бази даних, які можуть створювати об'єкти бази даних, тільки до адміністраторів баз даних, особливо в робочому середовищі.

Замість видалення облікового запису СУБД може надавати можливість блокування входу до системи. Блокування облікового запису забороняє користувачеві доступ до СУБД, але воно насправді не видаляє його із системи. Обліковий запис може бути згодом розблокований, тим самим дозволяючи

доступ до сервера. Такий процес дуже корисний, якщо адміністратор хоче заборонити доступ для тих користувачів, які не змінили свій пароль входу в систему вчасно.

Деякі СУБД надають додаткові елементи керування та параметри для облікових записів та паролів. Наприклад, деякі СУБД надають параметри профілю паролів, які можуть бути використані для обмеження наступного:

• кількість невдалих спроб входу до системи до блокування облікового запису.

• кількості днів, коли пароль діє, пільговий період для зміни пароля, що минув.

• кількість днів, коли обліковий запис може залишатися блокованим при закінченні терміну дії пароля.

• можливість повторного використання паролів (кількість днів, перш ніж пароль може бути використаний повторно, та максимальна кількість разів, коли пароль може бути використаний повторно).

Авторизація

Як тільки користувач отримує право доступу до СУБД, йому можуть автоматично надаватися різні інші привілеї, пов'язані з його ідентифікатором.

Авторизація – надання прав (або привілеїв), що дозволяють їх власнику мати законний доступ до системи чи її об'єктів.

Засоби авторизації користувачів можуть бути вбудовані безпосередньо в програмне забезпечення та керувати не тільки наданими користувачам правами доступу до системи або об'єктів, але й набором операцій, які користувачі можуть виконувати з кожним об'єктом, який їм доступний. З цієї причини механізм авторизації часто інакше називають підсистемою управління доступом. Термін «власник» у визначенні може представляти користувача-людину або програму. Термін «об'єкт» може представляти таблицю даних, подання, додаток, процедуру або будь-який інший об'єкт, який може бути створений у рамках системи.

В даний час використовуються два основні підходи до організації систем

безпеки: виборчий (дискреційний) та обов'язковий (мандатний). Обидва забезпечують створення системи безпеки як БД загалом, так окремих її об'єктів - таблиць, правил, підмножин кортежів тощо. аж до конкретного значення деякого атрибуту у певному кортежі певного відношення.

Виборчий підхід у тому, що з кожним конкретним ID пов'язується певний набір об'єктів БД і з кожною парою (ID, об'єкт) - певний набір привілеїв. При цьому різні ID мають, як правило, і різні привілеї по відношенню до тих самих об'єктів. Тому виборчі схеми захисту виявляються дуже гнучкими. Їх легко змінювати за зміни інформаційної політики підприємства. Цей підхід широко використовується в комерційних СУБД і детально обговорюється нижче.

Обов'язковий підхід у тому, кожному об'єкту захисту присвоюється певний рівень класифікації, а кожному користувачеві - рівень допуску з тими самими градаціями. Ці рівні утворюють ієрархію. Наприклад, особливої важливості > цілком секретно > секретно > для службового користування > відкрито. Тепер можна сформулювати два дуже прості (і жорсткі) правила безпеки.

• користувач і має доступ до об'єкта j, тільки якщо його рівень допуску більший або дорівнює рівню класифікації об'єкта j;

• користувач і може модифікувати об'єкт j тільки якщо його рівень допуску дорівнює рівню класифікації об'єкта j.

Друге правило означає, що будь-яка інформація, записана користувачем і, автоматично отримує рівень класифікації, що дорівнює рівню допуску користувача і. Це необхідно для того, щоб запобігти запису, наприклад, секретних даних у файл з рівнем класифікації для службового користування або відкрито.

Обов'язковий підхід використовується спеціальними системами, призначеними для державних, військових та інших організацій із жорсткою структурою.

Усі виборчі системи забезпечення безпеки реалізують загальну концепцію, що ґрунтується на наступних поняттях:

• користувач – системний ID, від імені якого СУБД виконує певні дії над

певними об'єктами;

• об'єкт захисту - частина БД, яку поширюється дію конкретного правила безпеки; це може бути група відносин, окреме відношення, підмножини атрибутів тощо;

• привілей - дія над об'єктом захисту, який може бути скоєно від імені конкретного ID.

Інформація про привілеї ID зберігається у системному каталозі. Вона використовується системою для ухвалення рішення щодо виконання запитаних ID операцій над даними. При цьому діє принцип: заборонено все, що явно не дозволено.

Виділяють два типи привілеїв:

• системні привілеї – права на створення та модифікацію об'єктів СУБД (користувачів, іменованих відносин, правил тощо);

• об'єктні привілеї - права використання об'єктів у операціях маніпулювання даними.

Власником бази даних є Адміністратор БД (АБД). Йому надано всі системні та об'єктні привілеї. Зокрема, він має право реєстрації нових користувачів та надання їм привілеїв як системних, так і об'єктних. Користувач, що має системні привілеї, є власником усіх створених ним об'єктів, має по відношенню до них усі привілеї та може надавати їх повністю або частково іншим користувачам.

Керування тим, які користувачі можуть отримувати доступ до яких об'єктів та команд, здійснюється за допомогою операторів GRANT та REVOKE:

• GRANT видає дозвіл користувачеві бази даних;

• REVOKE скасовує дозвіл, виданий користувачеві бази даних.

Оператор GRANT видається з двома супроводжуючими списками: список прав доступу, наданих списку користувачів. Для того, щоб використовувати оператор GRANT, користувач повинен бути власником об'єкта бази даних, йому має бути надано групове право доступу вищого рівня або видано WITH GRANT OPTION при наданні права доступу.

Синтаксис REVOKE є зворотним синтаксисом GRANT. Крім того, права доступу будуть автоматично скасовані СУБД у разі видалення об'єкта бази даних.

Типи прав доступу

Існують різні типи прав доступу, які можуть бути видані та скасовані для користувачів бази даних. Кожна СУБД надає певні основні типи прав доступу, такі як здатність отримувати доступ до даних, створювати об'єкти баз даних та виконувати системні функції. Кожна СУБД має додаткові типи прав доступу, залежно від властивостей, які вона підтримує.

Наступні типи прав доступу зазвичай надаються сучасними СУБД:

• До таблиці: керування тим, хто може отримувати доступ та модифікувати дані в межах таблиць.

• До об'єкта бази даних: керування тим, хто може створювати нові об'єкти баз даних та видаляти наявні об'єкти.

• До системи: управління тим, хто може виконувати певні типи системної діяльності.

• До програми: керування тим, хто може створювати, модифікувати та використовувати програми бази даних.

• До процедури, що зберігається: для керування тим, хто може виконувати певні функції та процедури, що зберігаються.

Видача прав доступу до таблиць.

Права доступу до таблиць надаються, щоб дозволити користувачам отримувати доступ до таблиць, переглядів та стовпців у межах таблиць та переглядів. Наступні права доступу можуть надаватися для таблиць та переглядів:

• SELECT: щоб дозволити користувачеві вибирати з цієї таблиці/перегляду.

• INSERT: щоб дозволити користувачеві вставляти рядки в таблицю/перегляд.

• UPDATE: щоб дозволити користувачеві оновлювати цю

таблицю/перегляд.

• DELETE: щоб дозволити користувачеві видаляти рядки цієї таблиці/перегляду.

• ALL: щоб дозволити користувачеві вибирати, вставляти, оновлювати та видаляти за допомогою цієї таблиці/перегляду.

Наприклад, щоб дозволити користувачеві user видалити рядки з таблиці table1, наступний оператор може бути створений:

GRANT DELETE on table1 to user;

Деякі права доступу до таблиць можуть визначатися лише на рівні шпальти. Виконання цього може бути кращим, коли певним користувачам має бути дозволено модифікувати певні стовпці таблиці, але не інші стовпці. Привілеї SELECT та UPDATE можуть видаватися для певних стовпців. Наприклад, щоб дозволити користувачеві user оновити тільки стовпець date таблиці table1, наступний оператор може бути виданий:

GRANT UPDATE on table1 (date) to user;

Зазвичай адміністратор баз даних видає права доступу до таблиць програмістам серед тестування з метою розробки. Крім того, програмісти та кінцеві користувачі можуть вимагати прав доступу до робочих таблиць для виконання певних завдань. Тим не менш, в основному доступ до робочих баз даних повинен контролюватись за допомогою прав доступу до програм і процедур, що зберігаються замість безпосереднього права доступу до таблиць.

Видача прав доступу до об'єктів баз даних

Права доступу до об'єктів баз даних управляють тим, яким користувачам надано дозвіл створювати структури баз даних, фактичні, права доступу, які можуть бути надані, залежать від СУБД та типів об'єктів баз даних, що підтримуються. Зазвичай СУБД надає можливості видачі прав доступу CREATE для кожного типу об'єктів баз даних, включаючи бази даних, табличні області, таблиці, індекси, тригери, значення за промовчанням та типи даних, що визначаються користувачем.

Наприклад, щоб дозволити користувачам user1 та user2 створювати

таблиці та індекси, наступний оператор може бути виданий:

GRANT CREATE table, CREATE index TO user1, user2;

Можливість створення об'єктів баз даних зазвичай закріплюється за адміністраторами бази даних. Якщо ці права доступу надаються іншим, кількість існуючих об'єктів баз даних може бути дуже важкою для управління. Крім того, буде досить важко простежити, які об'єкти бази даних дійсно використовуються, а які були створені, а потім покинуті. З цих причин адміністратор баз даних повинен залишати це право доступу за собою лише з рідкісними винятками для системних адміністраторів або дуже кваліфікованих розробників.

Видача прав доступу до системи

Права доступу до системи керують тим, які користувачі можуть використовувати певні властивості СУБД та виконувати певні команди СУБД. Можливі права доступу до системи змінюються в залежності від СУБД, але можуть включати здатність архівувати журнали реєстрації бази даних, відключати та перезапускати сервер баз даних, запускати відстеження, керувати постійною пам'яттю та кешами баз даних.

Права доступу до системи не можуть надаватися на рівні бази даних. Права доступу до системи надаються на системному рівні у СУБД. Наприклад, щоб дозволити користувачеві user запустити відстеження продуктивності, може бути заданий наступний оператор:

GRANT TRACE TO user

Права доступу до системи повинні видаватися з обережністю та зазвичай резервуються за адміністратором баз даних та системним адміністратором.

Видача прав доступу до програм та процедур

Надання права доступу EXECUTE дає користувачеві дозвіл виконувати програму або процедуру, що зберігається. Наприклад, щоб дозволити користувачам user1 і user2 виконати процедуру, що зберігається під назвою proc1, наступний оператор може бути виданий:

GRANT EXECUTE on proc1 TO user1, user2;

Надання користувачам прав доступу до програм та процедур легше в

управлінні, ніж надання прав доступу до окремих таблиць та стовпців. Процедурна логіка у програмі та процедурі регулює те, які певні таблиці та стовпці можуть бути модифіковані. Крім того, адміністратор баз даних може краще підтримувати цілісність робочих даних, якщо єдиний спосіб, яким вони можуть бути змінені, це програмно.

Видача PUBLIC

Як альтернатива надання доступу користувачеві бази даних адміністратор баз даних може вирішити надавати певну авторизацію PUBLIC. При видачі авторизації PUBLIC СУБД дозволяє кожному, хто має право доступу до СУБД, отримувати це право доступу. Право доступу PUBLIC не може бути видане з WITH GRANT OPTION, оскільки кожен із користувачів входить до складу PUBLIC.

Наприклад, щоб надати кожному право доступу видаляти рядки з таблиці table1, наступний оператор може бути виданий:

GRANT DELETE ON table1 TO PUBLIC;

Адміністрування безпеки може бути складним завданням. Використання права доступу PUBLIC для дозволу необмеженого доступу до певних об'єктів бази даних та ресурсів часто здається легшим, ніж спеціально контрольований доступ користувача. Тим не менш, адміністратори баз даних повинні бути обережними при видачі будь-яких прав доступу PUBLIC.

Щоразу, коли надається право доступу PUBLIC, адміністратор баз даних втрачає управління над цим об'єктом бази даних чи ресурсом - будь-який користувач може отримувати доступ або використовувати об'єкт чи ресурс відповідно до визначення оператора GRANT. Неминуче користувачі зловживатимуть ресурсами PUBLIC, і отримати право контролю над ресурсом знову буде дуже важко. Найкраще зарезервувати використання права доступу PUBLIC для декількох об'єктів і ресурсів баз даних, які повинні бути доступні кожному.

Рольова та групова авторизація

На додаток до прав доступу, що видаються окремим користувачам, СУБД

може надавати можливість призначення:

• певних прав доступу ролі, яка потім видається іншим;

• певних вбудованих груп доступу користувачам.

Термінологія не дотримується строго у всіх основних СУБД – деякі СУБД називають ролі групами та навпаки. Адміністратору баз даних необхідно розуміти, як кожна СУБД реалізує ролі та групи та як кожна з цих властивостей може бути використана для спрощення адміністрування безпеки бази даних.

Ролі

Як тільки вона визначена, роль може бути використана для видачі одного або кількох заздалегідь встановлених прав доступу користувачеві. Роль сутнісно є набором прав доступу. Адміністратор баз даних може створити роль та призначити певні права доступу для цієї ролі. Потім роль може бути присвоєна одному або декільком користувачам. Адміністрування безпеки бази даних таким чином спрощується. Наприклад, розглянемо таку послідовність операторів:

GRANT SELECT, INSERT, UPDATE, DELETE ON table1 TO programmer;

GRANT SELECT, INSERT, UPDATE, DELETE ON table2 TO programmer;

GRANT EXECUTE ON proc1 TO programmer

GRANT programmer TO user1

Цей сценарій створює нову роль під назвою programmer, видає права доступу до певних таблиць (table1,table2) і процедур (proc1) для ролі, а потім надає користувачеві user1 роль programmer. p align="justify"> Роль programmer може бути призначена і іншим користувачам, а адміністратору баз даних не потрібно буде пам'ятати про видачу для кожного з них окремого оператора GRANT, оскільки їм вже була присвоєна роль programmer.

Групи

Право доступу рівня групи схоже з участю. Проте кожна СУБД надає вбудовані групи, які не можуть бути змінені. Кожна СУБД реалізує безпеку бази даних лише на рівні групи у різний спосіб, під різними назвами і з різними правами доступу. Існують деякі подібні елементи у всіх СУБД [8,14]. Наступні

групи є загальними для основних СУБД:

• Системний адміністратор. Іноді скорочується як SA або SYSADM, група системного адміністратора є найпотужнішою в СУБД. Користувач, якому надається право доступу рівня системного адміністратора, зазвичай може виконувати всі команди бази даних і отримувати доступ до всіх баз даних та об'єктів. Системний адміністратор зазвичай відповідає за встановлення СУБД і вважається власником системних ресурсів і таблиць системного каталогу.

• Адміністратор бази даних. Іноді, що скорочується як DBADM або DBA, група адміністратора баз даних дає всі права доступу до певної бази даних, плюс здатність отримувати доступ, але не модифікувати дані в таблицях у межах цієї бази даних. Користувачі, яким присвоєно право доступу рівня адміністратора баз даних, можуть видаляти або змінювати будь-які об'єкти в базі даних (табличні області, таблиці та індекси).

• Обслуговування бази даних. Іноді, що скорочується як DBMAINT, група обслуговування баз даних включає певні права доступу до бази даних для обслуговування об'єктів баз даних (наприклад, здатність запускати утиліти і видавати команди). Подібно до групи адміністраторів баз даних, право доступу рівня DBMAINT надається на основі бази даних за базою даних.

• Адміністратор безпеки. Роль адміністратора безпеки має набір прав доступу, що дозволяє видачу та скасування безпеки бази даних у СУБД. Будь-яка діяльність, пов'язана з безпекою бази даних, може виконуватися адміністратором безпеки, включаючи адміністрування реєстраційних імен та паролів, аудит, конфігурування властивостей безпеки, а також видачу GRANT та REVOKE. Іншою загальною назвою ролі адміністратора безпеки є SSO.

• Контроль операцій. Іноді звана OPER або SYSOPR, роль контролю операцій має право виконувати завдання працюючої бази даних, такі, як резервування та відновлення або завершення завдань, що вийшли з-під контролю.

Організація повинна обмежувати кількість користувачів, яким надається роль системного адміністратора або право доступу до рівня групи. Користувач із

можливостями системного адміністратора має дуже потужну владу. Тільки корпоративні адміністратори баз даних та системні програмісти мають отримувати права доступу такого рівня. Кінцеві користувачі, менеджери та персонал розробки додатків не потребують прав доступу системного адміністратора для виконання своєї роботи.

Безпека рівня групи та каскадні REVOKE

Коли права доступу скасовуються, СУБД повинна вирішити, чи необхідні додаткові REVOKE, ґрунтуючись на правах доступу, що скасовуються. Коли одне скасування призводить до того, що СУБД скасовує додаткові права права доступу, це називається каскадними REVOKE. Залежно від групи деякі користувачі, яким було призначено права доступу до рівня групи, можуть видавати права доступу іншим користувачам. Якщо право доступу рівня групи скасовується для цього користувача, будь-які права доступу, видані цим користувачем, також будуть скасовані. Це подібно до каскадних REVOKE, які відбуваються в результаті вибору WITH GRANT OPTION.

Перед скасуванням права доступу рівня групи для користувача необхідно переконатися, що встановлено вплив каскадних REVOKE і бути готовим до того, щоб встановити повторно необхідні права доступу, які будуть видалені через ефект каскадування.

Найкращий принцип управління привілеями – звести їх до мінімуму. У загальному випадку ідея надавати привілеї лише тим користувачам, яким вони необхідні, хороша, але до надання деяких з них слід ставитись особливо уважно:

• Привілей GRANT дозволяє користувачам передавати свої привілеї іншим користувачам. Два користувача з різними привілеями, маючи привілей GRANT, здатні об'єднати свої привілеї.

• Привілей ALTER може бути використаний для перейменування таблиць та руйнування таким чином усієї системи привілеїв.

• Привілей FILE може використовуватися зловмисно для зчитування будь-якого засекреченого файлу, що зберігається на сервері, в таблицю бази даних, до вмісту якої можна отримати доступ за допомогою команди SELECT. Це

стосується і вмісту всіх баз даних, які знаходяться під керуванням сервера

Інші механізми захисту баз даних

Сучасні реляційні СУБД підтримують безліч можливостей та якостей, які можуть допомогти у забезпеченні безпеки даних. Деякі з цих можливостей не є у своїй основі якістю безпеки. Наприклад, перегляди та процедури, що зберігаються, можуть використовуватися для цілей безпеки, хоча це і не основне їх призначення.

Подання (перегляди)

Більшість операцій забезпечення безпеки баз даних виконується за допомогою вбудованої безпеки СУБД. Проте, можна спростити деякі аспекти безпеки бази даних, створюючи перегляди захисту даних.

Уявлення – це динамічний результат однієї чи кількох реляційних операцій із базовими відносинами з єдиною метою створення деякого іншого відносини. Подання є віртуальним ставленням, якого реально в базі даних не існує, але яке створюється на вимогу окремого користувача в момент надходження цієї вимоги.

Механізм уявлення є потужним і гнучкий інструмент організації захисту даних, що дозволяє приховувати від певних користувачів деякі частини бази даних. В результаті користувачі не матимуть жодних відомостей про існування будь-яких атрибутів або рядків даних, які недоступні через уявлення, що знаходяться у їхньому розпорядженні. Подання може бути визначене на базі кількох таблиць, після чого користувачі будуть надані необхідні привілеї доступу до цього подання, але не до базових таблиць. У разі використання уявлення є більш жорсткий механізм контролю доступу, ніж звичайне надання користувачеві тих чи інших прав доступу до базовим таблицям.

Наприклад, щоб виключити секретну інформацію, що зберігається в третьому стовпці таблиці table1, може бути створений наступний перегляд forall:

```
CREATE VIEW forall
AS
SELECT colum1,colum2 FROM table1
```

Як тільки перегляд був створений, і користувачеві було надано право доступу SELECT для даного подання, лише інформація, вказана у перегляді, може бути вилучена.

Використання процедур, що зберігаються для забезпечення безпеки

Процедури, що зберігаються, можуть використовуватися для забезпечення додаткового рівня безпеки. Право доступу на виконання процедури, що зберігається, має бути видане або скасоване явно, незалежно від безпеки, що реалізується в базових таблицях, що лежать в основі.

Можуть бути кодовані процедури, що зберігаються, які отримують доступ тільки до підмножин даних на рівні рядків або стовпців. Здатність виконувати ці збережені процедури може потім надаватися користувачам. Якщо жодних прав доступу до базових таблиць, що лежать в основі, не надано, користувачі можуть отримувати доступ до даних тільки шляхом виконання процедури, що зберігається, тим самим забезпечуючи необхідну безпеку.

На додаток до надання рівня безпеки, цей метод може забезпечити більш високу продуктивність, якщо алгоритми в процедурі кодовані правильно.

Резервне копіювання та відновлення

Резервне копіювання – процедура, що періодично виконується, отримання копії бази даних та її файлу журналу (а також, можливо, програм) на носії, що зберігається окремо від системи. Будь-яка сучасна СУБД має надавати кошти резервного копіювання, що дозволяють відновлювати базу даних у разі її руйнування. Крім того, рекомендується створювати резервні копії бази даних та її файлу журналу з деякою встановленою періодичністю, а також організовувати збереження створених копій у місцях, забезпечених необхідним захистом. У разі відмови, внаслідок якої база даних стає непридатною для подальшої експлуатації, резервна копія та зафіксована у файлі журналу оперативна інформація використовуються для відновлення бази даних до останнього узгодженого стану.

Ведення журналу - процедура створення та обслуговування файлу журналу, що містить відомості про всі зміни, внесені в базу даних з моменту

створення останньої резервної копії, і призначеного для забезпечення ефективного відновлення системи у разі її відмови.

СУБД має надавати засоби ведення системного журналу, в якому фіксуватимуться відомості про всі зміни стану бази даних та хід виконання поточних транзакцій, що необхідно для ефективного відновлення бази даних у разі відмови. Переваги використання такого журналу полягають у тому, що у разі порушення роботи або відмови СУБД базу даних можна буде відновити до останнього відомого узгодженого стану, скориставшись останньою створеною резервною копією бази даних та оперативною інформацією, що міститься у файлі журналу. Якщо в системі, що відмовила, функція ведення системного журналу не використовувалася, базу даних можна буде відновити тільки до того стану, який було зафіксовано в останній створеної резервної копії. Усі зміни, які були внесені до бази даних після створення останньої резервної копії, виявляться втраченими.

Контрольна точка – момент синхронізації між станом бази даних та станом журналу виконання транзакцій. У цей момент усі буфери примусово вивантажуються на пристрої вторинної пам'яті.

Сучасні СУБД зазвичай надають засоби створення контрольних точок, що дозволяють зафіксувати в базі даних серію останніх виконаних змін. Механізм створення контрольних точок може використовуватися разом із веденням системного журналу, що дозволить підвищити ефективність процесу відновлення. У момент створення контрольної точки СУБД виконує дії, що забезпечують запис на диск всіх даних, що зберігалися в основній пам'яті машини, а також розміщення файлу журналу спеціального запису контрольної точки.

Процедури, що регламентують процеси створення резервних копій, визначаються типом і розмірами бази даних, що експлуатується, а також тим набором відповідних інструментів, який надається використовуваною СУБД. Ці процедури повинні включати необхідні етапи, на яких безпосередньо виконуватиметься створення резервної копії. Великі бази даних можуть

повністю копіюватися раз на тиждень або навіть раз на місяць, але при цьому слід організувати обов'язкове копіювання, що виконується з вищою частотою. День та годину виконання резервного копіювання повинні встановлюватись відповідальними особами.

У процедурах копіювання також може вказуватися, які частини системи (наприклад, прикладні програми), крім самих даних, повинні підлягати копіюванню. Залежно від частоти внесення змін до системи, протягом однієї доби може виконуватися кілька копіювань; створені копії повинні розміщуватись у безпечному місці. Місце зберігання останніх копій повинно бути обладнане як мінімум шафами, що не згорають. Крім того, бажано використовувати деяке зовнішнє сховище, в яке буде поміщатися другий екземпляр створених копій. Всі згадані деталі повинні знайти своє чітке відображення в розроблених процедурах резервного копіювання, які повинні виконуватися обслуговуючим персоналом.

Як і процедури копіювання, процедури відновлення також мають бути ретельно продумані та опрацьовані. Те, які саме процедури відновлення будуть виконуватися, має визначатися типом відмови (руйнування носія, відмова програмного забезпечення або відмова обладнання системи). Крім того, процедурами відновлення повинні враховуватися особливості методів відновлення, прийнятих у СУБД, що використовується. У будь-якому випадку розроблені процедури відновлення повинні бути ретельно протестовані, оскільки необхідно отримати повну гарантію, що вони працюють правильно ще до того, як відбудеться реальна відмова системи. В ідеалі процедури відновлення повинні регулярно тестуватися з деяким встановленим інтервалом.

Аудит

Аудит є засобом СУБД, який дозволяє адміністраторам баз даних відстежувати використання ресурсів бази даних та прав доступу. Коли аудит включений, СУБД створює контрольний журнал операцій бази даних. Кожна операція бази даних, що перевіряється, видає контрольну інформацію, включаючи те, що вплинуло на об'єкт бази даних, хто виконав операцію і коли.

Залежно від ступеня підтримки аудиту СУБД може бути зроблено запис того, які дані було змінено. Відстеження того, хто і що і з якими даними робить, важливо, оскільки існує безліч загроз для безпеки ваших даних.

Слід мати на увазі, що аудит відстежує все, що конкретний користувач зробив, як тільки доступ було дозволено. Аудит здійснюється після здійснення дії, він не робить чогось; щоб заборонити доступ. Контрольні відстеження допомагають зберегти цілісність даних, дозволяючи виявити порушення безпеки, також звані виявлення вторгнень. Система, що зазнала аудиту, може послужити як засіб стримування користувачів, що фальсифікують дані, оскільки вона допомагає ідентифікувати таємних шпигунів.

Контрольний журнал може бути корисним у багатьох ситуаціях. Практичні методи ведення бізнесу та служби безпеки вашої компанії можуть диктувати необхідність всебічного відстеження кожної зміни даних аж до ініціатора дії. Можливо, державні норми та постанови вимагають, щоб ваша організація аналізувала доступ до даних та створювала регулярні звіти. Вам, можливо, потрібно створювати докладні звіти на постійній основі, або вам просто потрібна можливість знайти основну причину проблем з цілісністю даних час від часу. Аудит корисний у будь-якому випадку.

Типовий засіб аудиту дозволяє відслідковувати операції на різних рівнях у межах СУБД - наприклад, на рівні баз даних, об'єктів баз даних та рівні користувачів. Однією з найбільших проблем при використанні коштів аудиту СУБД є зниження продуктивності. Створювані контрольні записи повинні бути докладними достатньо, щоб описувати станів баз даних як до, так і після зміни. Однак фіксація такої кількості інформації, особливо в завантаженій системі, може знизити продуктивність. Більше того, цей контрольний запис повинен десь зберігатися, що проблематично, коли відбувається велика кількість змін. Тому більшість засобів аудиту дозволяє вибіркове створення записів ау-диту для мінімізації проблем з продуктивністю та зберіганням.

Хоча кожна СУБД пропонує різні можливості проведення аудиту, існують деякі загальні моменти, які можуть відстежуватися засобами аудиту всіх СУБД.

Вони включають:

• спроби входу та виходу (як успішні, так і невдалі);

• перезапуск сервера баз даних;

• команди, які видаються користувачами з правами доступу системного адміністратора;

• спроби порушення цілісності (де дані, що змінюються або вставляються, не відповідають посилальному, унікальному або контрольному обмеженню);

• операції SELECT, INSERT, UPDATE та DELETE;

• виконання процедур, що зберігаються;

• невдалі спроби доступу до бази даних або таблиці (неуспішні авторизації);

• зміни таблиць системного каталогу;

• операції рівня рядків.

Кожна СУБД надає різноманітні засоби перегляду даних аудиту. Відформатовані звіти та інструменти графічної звітності, які зчитують та подають інформацію аудиту раціональним чином, можуть полегшити ідентифікацію проблем безпеки серед багатьох зареєстрованих операцій бази даних.

Шифрування

Якщо в системі з базою даних міститься дуже важлива конфіденційна інформація, то є сенс закодувати її з метою запобігання можливої загрози несанкціонованого доступу із зовнішнього боку (стосовно СУБД). Деякі СУБД включають засоби шифрування, призначені для використання в подібних цілях.

Шифрування – кодування даних із використанням спеціального алгоритму, у результаті дані стають недоступними читання будь-який програмою, яка має ключа дешифрування.

Підпрограми СУБД, що включають засоби шифрування, забезпечують санкціонований доступ до даних (після їх декодування), хоча це буде пов'язано з деяким зниженням продуктивності, викликаним необхідністю перекодування. Шифрування також може використовуватися для захисту даних при їх передачі

лініями зв'язку. Існує безліч різних технологій кодування даних з метою приховування інформації, що передається, причому одні з них називають незворотними, а інші оборотними. Необоротні методи, як і випливає з їхньої назви, не дозволяють встановити вихідні дані, хоча останні можуть використовуватися для збирання достовірної статистичної інформації. Оборотні технології використовуються частіше. Для організації захищеної передачі даних незахищеними мережами повинні використовуватися системи шифрування, що включають такі компоненти:

• ключ шифрування, який призначений для шифрування вихідних даних (звичайного тексту);

• алгоритм шифрування, який описує, як за допомогою ключа шифрування перетворити звичайний текст на шифротекст;

• ключ дешифрування призначений для дешифрування шифротексту;

• алгоритм дешифрування, який описує, як за допомогою ключа дешифрування перетворити шифротекст на вихідний звичайний текст.

Деякі системи шифрування, які називають симетричними, використовують один і той же ключ як для шифрування, так і для дешифрування, при цьому передбачається наявність захищених ліній зв'язку, призначених для обміну ключами.

Одна з найпоширеніших систем шифрування називається DES (Data Encryjption Standard) – у ній використовується стандартний алгоритм шифрування, розроблений фірмою IBM. У цій схемі для шифрування та дешифрування використовується той самий ключ, який повинен зберігатися в секреті, хоча сам алгоритм шифрування не є секретним. Цей алгоритм передбачає перетворення кожного 64-бітового блоку звичайного тексту за допомогою 56-бітового ключа шифрування. Система шифрування DES розцінюється як досить надійна далеко не всіма – деякі розробники вважають, що варто використовувати більш довге значення ключа.

Інший тип систем шифрування передбачає використання для шифрування та дешифрування повідомлень різних ключів – подібні системи прийнято

називати несиметричними. Прикладом такої системи є система з відкритим ключем, що передбачає використання двох ключів, один із яких є відкритим, а інший зберігається в секреті. Алгоритм шифрування також може бути відкритим, тому будь-який користувач, який бажає направити власнику ключів зашифроване повідомлення, може використовувати його відкритий ключ і відповідний алгоритм шифрування. Однак, дешифрувати це повідомлення зможе тільки той, хто знає парний закритий ключ шифрування. Системи шифрування з відкритим ключем можуть також використовуватися для відправки разом із повідомленням "цифрового підпису", що підтверджує, що це повідомлення було справді надіслано власником відкритого ключа. Найбільш популярною несиметричною системою шифрування є RSA (ініціали трьох розробників цього алгоритму). Як правило, симетричні алгоритми є швидкодіючішими, ніж несиметричні, проте на практиці обидві схеми часто застосовуються спільно, коли алгоритм з відкритим ключем використовується для шифрування випадковим чином згенерованого ключа шифрування, а вже цей випадковий ключ – для шифрування самого повідомлення із застосуванням деякого симетричний алгоритм.

Встановлення нового прикладного програмного забезпечення

Нові додатки, розроблені власними силами або сторонніми організаціями, обов'язково слід ретельно протестувати, перш ніж приймати рішення про їх розгортання та передачу в експлуатацію. Якщо рівень тестування буде недостатнім, суттєво зростає ризик руйнування бази даних. Слід вважати доброю практикою виконання резервного копіювання бази даних безпосередньо перед здаванням нового програмного забезпечення в експлуатацію. З іншого боку, у період експлуатації нового докладання обов'язково слід організувати ретельне спостереження функціонуванням системи.

Встановлення чи модернізація системного програмного забезпечення

В обов'язки адміністратора бази даних входить стеження за виходом нових версій програмного забезпечення, що використовується, і виконання модернізації СУБД при надходженні від розробника чергових пакетів змін. У

деяких випадках зміни, що вносяться, виявляються зовсім незначними і стосуються тільки невеликої частини модулів системи, але навіть дуже незначні зміни можуть бути важливими в тих випадках, коли розробники виправили якусь помилку (bug), яка могла призвести до краху всієї системи. А можливі ситуації, коли буде потрібна повна ревізія всієї встановленої системи. Як правило, кожен пакет змін, що надходить, супроводжується друкованою або інтерактивною документацією, що містить докладніше відомості про сутність змін та їх призначення. Багато хто схильний вважати, що установки будь-якого з пакетів модернізації продовження нормального функціонування існуючих баз даних і прикладного програмного забезпечення автоматично гарантується. Зі зростанням інтенсивності використання бази даних та збільшенням обсягу супровідної документації до пакету ця переконаність зміцнюється. Однак забезпечення захисту даних та додатків має переважну важливість, і це слід враховувати. Жодні зміни та модернізації в жодному разі не повинні вноситися до системи без попередньої оцінки їхнього можливого впливу на наявні дані та програмне забезпечення.

Результатом ознайомлення із супровідною документацією пакета має стати план дій щодо його встановлення. У цьому плані мають бути відображені будь-які зміни, які можуть вплинути на базу даних та додатки, а також запропоновані рішення щодо їх реалізації. Які зміни не знадобляться, всі вони повинні бути враховані і для кожного з них має бути дано оцінку часу виконання з урахуванням обсягу всього наявного програмного забезпечення. Головне завдання адміністратора – забезпечити плавний та безболісний перехід від старої версії системи до нової.

У функціонуючій системі, яка повинна бути постійно доступна протягом усього робочого часу, установки будь-яких пакетів модернізації зазвичай виконуються у позаробочий час – наприклад, у вихідні. Безпосередньо перед виконанням модернізації має бути зроблено повну резервну копію існуючої системи - на випадок можливої відмови. Потім виконується встановлення пакета модернізації, а дані і програми вносяться всі необхідні зміни, супроводжувані

необхідними процедурами тестування. Тільки після завершення зазначених процедур система може бути запущена в роботу з використанням реальних даних.

Зміни, що вносяться в програми СУБД в результаті модернізації, можуть вплинути на будь-які прикладні програми, створені різними програмістами, тому список необхідних змін повинен бути або розісланий всім зацікавленим особам, або відкритий для загального доступу. Деякі з змін можуть представляти особливий інтерес, якщо вони пов'язані, наприклад, з виправленням помічених раніше помилок і дозволяють видалити штучні способи їх обходу, що використовувалися до цього. Крім того, бажано, щоб супровідна документація включала список відомих помилок із зазначенням шляхів їх обходу ("латок"), що використовувалися. Ці відомості також мають бути розіслані всім заінтересованим особам або зроблені загальнодоступними.

2.3. Особливості безпеки даних у СУБД MySQL

MySQL – дуже швидкий, багатопотоковий (може легко використовувати багато процесів), багатоплатформний (доступний на більш ніж 20 відомих платформ, включаючи більшість дистрибутивів Linux, Mac OS X, UNIX та Microsoft Windows), багатокористувацький та підтримуючий SQL сервер баз даних. MySQL є free software. Він ліцензується за GNU GENERAL PUBLIC LICENSE.

Дистрибутив MySQL існує як у двійковому вигляді, так і у вигляді вихідного тексту. Можна знайти двійковий дистрибутив для своєї платформи, чи взяти дистрибутив з вихідним кодом і скомпілювати сервер "під себе".

MySQL є системою "клієнт-сервер", що складається з багатопоточного (для кожного з'єднання клієнта з сервером створюється свій потік) SQL-сервера, який підтримує різні функції, кількох різних клієнтських програм та бібліотек, адміністративних інструментальних засобів та кількох інтерфейсів програмування.

СУБД MySQL працює за протоколами TCP/IP, як та інші Internet-сервіси. З'єднання різняться на ім'я вузла і номер порту. За замовчуванням

використовується порт 3306, але це параметр, що конфігурується. Цей порт постійно прослуховується сервером MySQL (демон mysqld). За сеансом закріплюються два порти: один використовуватиметься відправлення даних, інший – їхнього приема. Під час сеансу клієнт надсилає серверу команди, які мають вигляд інструкцій SQL. У відповідь деякі інструкції сервер повертає дані, а клієнт форматує їх відображення на екрані.

Програма MySQL виконується у вигляді системного сервісу, тому слід враховувати засоби безпеки, що надаються операційною системою або мережевим програмним забезпеченням.

Використовувана в MySQL система безпеки для всіх підключень, запитів та інших операцій, які може намагатися виконати користувач, базується на списках контролю доступу ACLs (Access Control Lists). У списку управління доступом вказується, які інструкції дозволено виконувати тим чи іншим користувачам. Такий список можна пов'язати з користувачем, вузлом, базою даних, таблицею або стовпцем. Програма MySQL звіряє кожне звернення до бази даних з наявними списками управління доступом і визначає, чи є у користувача право виконати дія, що запитується.

У полях привілеїв вказуються привілеї, що надаються записом таблиці, тобто. те, які операції можна виконувати. Сервер формує повний опис привілеїв користувача, комбінуючи інформацію, що зберігається в різних таблицях привілеїв.

Користувачі ідентифікують себе на ім'я, пароль та адресу вузла. Імена користувача та паролі MySQL не пов'язані безпосередньо з іменами та паролями операційної системи. Просто більшість клієнтів MySQL за промовчанням бере ім'я користувача, яке було вказано під час реєстрації в системі. Це досить зручно, хоч і не є обов'язковим правилом.

Імена користувача та паролі можуть бути довжиною до 16 символів. Пароль дозволяється залишати порожнім. Це найнижчий рівень безпеки. Він допустимо лише в тому випадку, коли доступ обмежується за іншими критеріями, наприклад за адресою вузла.

Важливий принцип, який полягає в наступному: при збереженні непустих паролів з використанням операторів INSERT або UPDATE для їх шифрування повинна застосовуватися функція PASSWORD(). Це робиться тому, що в таблиці user паролі зберігаються у зашифрованому вигляді, а не як простий текст.

Якщо пароль задається оператором GRANT ... IDENTIFIED BY або команди mysqladmin password, немає необхідності використовувати функцію PASSWORD().

Починаючи з MySQL версії 4.0.2, можна обмежувати певні ресурси, що виділяються користувачам. До цієї версії єдиним можливим методом обмеження використання ресурсів сервера MySQL була установка пе-ременной запуску max_user_connections в значення, відмінне від нуля. Але цей метод діє тільки на глобальному рівні і не дозволяє керувати окремими користувачами. Він може представляти певний інтерес лише провайдерів послуг Internet.

На рівні окремого користувача тепер введено управління наступними трьома ресурсами:

• Кількість всіх запитів на годину: всі команди, які користувач може запускати.

• Кількість всіх оновлень за годину: будь-яка команда, яка змінює таблицю або базу даних.

• Кількість з'єднань за годину: нові з'єднання, відкриті за годину.

MySQL підтримує шифровані SSL з'єднання.

За умовчанням MySQL налаштований максимально швидку роботу і тому використовуються незашифровані з'єднання між клієнтом і сервером. Це означає, що переглядати всі дані, що передаються між клієнтом і сервером, може будь-хто. Насправді можна навіть змінювати дані під час передачі їх від клієнта до сервера і навпаки. Крім того, іноді виникає необхідність передати дійсно секретні дані через загальнодоступну мережу - у таких випадках використання незашифрованих з'єднань просто неприйнятно.

У протоколі SSL використовуються різні алгоритми шифрування, що забезпечують безпеку для даних, що передаються через загальнодоступні

мережі. Цей протокол містить засоби, що дозволяють виявляти будь-які зміни, втрати та повтори даних. У протоколі SSL також застосовуються алгоритми для проведення ідентифікації за допомогою стандарту X509.

Стандарт X509 дозволяє робити ідентифікацію в Internet. Найчастіше він використовується у додатках електронної комерції. Спрощено схема його застосування виглядає так: існує певний центр сертифікації, який призначає електронні сертифікати всім, кому вони потрібні. Сертифікати ґрунтуються на асиметричних алгоритмах шифрування, що містять два ключі - публічний і секретний. Власник сертифіката може підтвердити свою особу, пред'явивши свій сертифікат іншій стороні. Сертифікат складається із публічного ключа власника. Будь-які дані, зашифровані за допомогою цього публічного ключа, можуть бути розшифровані тільки за допомогою відповідного секретного ключа, який знаходиться у власника сертифіката.

2.4. Особливості безпеки даних у СУБД ORACLE

Oracle – потужна, багатоплатформова, що підтримує архітектуру з кількома процесами СУБД, працює на сьогодні з найбільшими базами даних, потенційного розміру до кількох терабайт, що підтримує розподілену обробку даних та надає повний контроль розподілу простору. ORACLE підтримує велику кількість користувачів, що одночасно виконують різноманітні додатки, які оперують одними і тими ж даними за допомогою механізму транзакцій. Він мінімізує суперництво за дані та гарантує узгодженість даних.

ORACLE дозволяє писати процедури, які виконуються автоматично в результаті оновлення, вставки або видалення з таблиці. Такі процедури називаються тригерами бази даних. Тригери бази даних можуть використовуватися найрізноманітнішими способами для інформаційного управління базою даних. Наприклад, їх можна використовувати для автоматизації генерації даних, аудиту модифікацій даних, введення в дію комплексних обмежень цілісності або організації складних процедур забезпечення захисту.

Програмне забезпечення ORACLE переносимо між різними операційними системами і однаково у всіх системах. Програми, що розробляються для ORACLE, можуть бути перенесені в будь-яку операційну систему з мінімумом модифікацій або взагалі без таких.

СУБД складається з двох складових: база даних та екземпляр (конкретна реалізація системи). База даних складається з фізичних файлів, що зберігаються в системі, та з логічних частин (наприклад, схема БД). Ці файли можуть бути різними. Примірник – це спосіб доступу до даних, що складається з процесів та системної пам'яті.

Фізична структура бази даних ORACLE визначається файлами операційної системи, з яких складається база даних. Кожна база даних ORACLE складається з файлів трьох типів: одного або декількох файлів даних, двох або більше файлів журналу повторення роботи, що використовується для відновлення системи у разі збою, і одного або кількох керуючих файлів. Файли бази даних надають дійсну фізичну пам'ять для інформації бази даних. У керуючому файлі записується фізична структура бази даних. Зокрема, цей файл містить таку інформацію:

• ім'я бази даних;

• імена та розташування файлів даних та файлів журналу повторення цієї бази даних;

• позначку часу створення бази даних.

При кожному запуску інстанції бази даних ORACLE її керуючий файл використовується для того, щоб ідентифікувати базу даних та файли журналу повторення, які повинні бути відкриті для продовження роботи бази даних. Коли фізичний склад бази даних змінюється (наприклад, створюється новий файл даних або файл журналу), ORACLE автоматично модифікує керуючий файл, щоб відобразити цю зміну. Керуючий файл бази даних використовується також у випадках, коли потрібне відновлення бази даних.

Кожна база даних ORACLE має словник даних. Він являє собою набір таблиць і оглядів, що використовуються як тільки представлення бази даних.

Наприклад, у словнику даних зберігається інформація про логічну і фізичну структуру бази даних. Крім цієї важливої інформації, словник даних зберігає таку інформацію:

• про дійсних користувачів бази даних ORACLE

• про обмеження цілісності, визначені для таблиць бази даних

• про те, скільки простору розподілено для кожного об'єкта схеми, та скільки з нього використовується

Словник даних створюється під час створення самої бази даних. Щоб точно відображати стан бази даних у будь-який момент, словник даних автоматично оновлюється ORACLE у відповідь на спеціальні дії (такі як зміна структури бази даних). Словник даних критичний для працездатності бази даних, бо на ньому грунтується реєстрація, верифікація та управління поточною роботою. Наприклад, під час операцій з базою даних ORACLE звертається до словника даних для перевірки того, що об'єкти схем існують і що користувачі мають до них відповідні права доступу [1].

Кожна база даних ORACLE має перелік імен користувачів. Щоб отримати доступ до бази даних, користувач повинен використовувати програму бази даних, і спробувати з'єднатися з базою даних, надавши дійсне ім'я користувача. З кожним ім'ям користувача пов'язаний пароль, щоб запобігти несанкціонованому доступу.

З кожним ім'ям користувача асоційована СХЕМА (SCHEMA), що має таке ж ім'я. SCHEMA (схема) - це логічна колекція об'єктів бази даних (таблиць, уявлень, послідовностей, синонімів, індексів, кластерів, процедур, функцій, пакетів та зв'язків баз даних). За замовчуванням кожен користувач бази даних створює і має доступ до всіх об'єктів у відповідній схемі.

Кожен користувач має домен захисту - набір властивостей, які визначають такі речі, як:

• дії (привілеї та ролі), доступні користувачеві;

• квоти табличних просторів (доступної дискової пам'яті);

• ліміти на системні ресурси (наприклад, час процесора) для даного

користувача.

Привілеї в СУБД ORACLE можуть бути поділені на дві різні категорії: системні привілеї та об'єктні привілеї [1].

Системні привілеї (SYSTEM_PRIVILEGE) дозволяють користувачам виконувати конкретну дію на рівні системи, або конкретну дію над конкретним типом об'єктів. Такі, наприклад, привілей створювати табличний простір або привілей видаляти рядки будь-якої таблиці в базі даних. Більшість системних привілеїв доступні лише адміністраторам і розробникам додатків, оскільки такі привілеї дуже потужні.

Якщо потрібно буде дати привілеї системного рівня кожному користувачеві Oracle (включаючи будь-яких користувачів Oracle, які можуть бути створені в майбутньому), можна дати системний привілей користувачу PUBLIC, що означає будь-якого користувача Oracle.

Привілеї системного рівня можуть бути надані користувачеві Oracle з використанням WITH ADMIN OPTION в кінці оператора GRANT. Це дозволяє користувачеві Oracle, який отримав такий системний привілей, передавати її будь-якому іншому користувачеві Oracle. По суті він стає ще одним адміністратором цього привілею.

Об'єктні привілеї дають змогу користувачам виконувати конкретні дії на конкретному об'єкті. Такий, наприклад, привілей видаляти рядки у зазначеній таблиці. Об'єктні привілеї призначаються користувачам, отже вони можуть використовувати програми бази даних до виконання конкретних завдань.

Якщо користувач Oracle володіє об'єктом на зразок таблиці, іншим користувачам Oracle може бути дозволено використовувати цей об'єкт шляхом надання їм одного або кількох привілеїв об'єктного рівня. Наприклад, якщо користувач user_1 має намір дозволити користувачеві user_2 використовувати його таблицю, але тільки для вставки та оновлення рядків за допомогою команд INSERT або UPDATE, то користувачеві user_2 можуть бути дані привілеї UPDATE і INSERT.

ORACLE дозволяє здійснювати вибірковий АУДИТ (реєстроване

відстеження) дій користувачів, щоб допомогти розслідувати випадки підозрілого використання бази даних. Аудит може виконуватися на трьох різних рівнях: аудит пропозицій, аудит привілеїв та аудит об'єктів.

Аудит пропозицій

Відстеження специфічних пропозицій SQL безвідносно до конкретних об'єктів. Аудит пропозицій може бути широким, відстежуючи всіх користувачів у системі, або може бути сфокусований лише обраних користувачів. Наприклад, можна відстежувати підключення та відключення від бази даних користувачів roma та pasha.

Аудит привілеїв

Відстеження використання потужних системних привілеїв безвідносно до конкретних об'єктів. Аудит привілеїв може бути широким, відстежуючи всіх користувачів у системі, або може бути сфокусований лише на вибраних користувачах.

Аудит об'єктів

Відстеження звернень до об'єктів схем безвідносно до користувачів. Аудит об'єктів відстежує пропозиції, що дозволяються привілеями об'єкта, такими як пропозиції SELECT або DELETE на даній таблиці.

Для всіх типів аудиту ORACLE допускає вибіркове відстеження успішних виконань пропозицій, безуспішних виконань, або тих, та інших. Це дозволяє відстежувати підозрілі пропозиції незалежно від того, чи мав користувач, який видав їх, відповідні привілеї видавати такі пропозиції.

Результати аудиту записуються у спеціальну таблицю, відому як аудиторський журнал (audit trail). Існують зумовлені огляди по цій таблиці, так що записи аудиторського журналу можуть бути легко вилучені [1].

2.5. Особливості безпеки даних у СУБД MS SQL Server

MSSQL Server – одна з найпотужніших комерційних СУБД з архітектурою «клієнт-сервер» (сервер за замовчуванням працює на порту 1433). Ця СУБД дозволяє задовольняти таким вимогам, що висуваються до систем розподіленої

обробки, як тиражування даних, паралельна обробка, підтримка великих баз даних на відносно недорогих апаратних платформах.

MSSQL Server має багатопоточну архітектуру, тобто. використовує лише один виконуваний файл, з кількома потоками виконання. Сервер бере на себе поділ часу між окремими потоками, іноді даючи перевагу деяким завданням перед іншими. Таким чином, відпадає необхідність у складному механізмі взаємодії процесів і знижує вимоги до ресурсів процесора та пам'яті. А починаючи з версії MS SQL Server 2000, можна на одній машині запускати кілька екземплярів SQL Server. У більш ранніх версіях можна було мати на одному сервері кілька баз даних, але в кожний момент часу тільки одна з них могла бути активною. Підтримка одночасної роботи декількох екземплярів дозволяє виконувати такі основоположні роботи, як супровід на одній і тій же фізичній машині двох незалежних середовищ SQL Server для команд, що розробляють і тестують. І, що більш важливо, ця властивість полегшує Microsoft підтримку SQL Server у повністю кластеризованому середовищі.

SQL Server використовує для зберігання баз даних набір файлів операційної системи, при цьому для кожної з них створюється власний файл. Декілька баз даних більше не можуть знаходитися в одному файлі. Існує кілька важливих переваг такого спрощення: файли можуть розростатися та скорочуватися, керування простором також стало простіше.

Всі дані та об'єкти бази даних, такі як таблиці, процедури, що зберігаються, тригери і уявлення, зберігаються в наступних файлах [6].

Крім того, можуть створюватися додаткові групи файлів для розміщення даних користувача.

При створенні бази даних всі файли, що входять до її складу, "обнуляються" (заповнюються нулями), щоб стерти всі дані, які залишилися на диску від раніше віддалених файлів. Хоча це призводить до збільшення тривалості створення БД, це позбавляє ОС необхідності очищення файлів при записі даних у них під час нормальної роботи з базою даних, що підвищує продуктивність системи.

База даних складається з одного або кількох файлів даних і одного або кількох файлів журналу. Файли даних можуть бути згруповані у визначені користувачем групи. Таблиці та індекси можуть бути поставлені у відповідність різним групам файлів для управління розміщенням даних на фізичних дисках.

"Просунуті" користувачі можуть з успіхом використовувати групи файлів для того, щоб розмістити індекси та таблиці певним чином. SQL Server може ефективно працювати і без групування файлів, і в більшості систем немає необхідності визначати власні групи файлів. У цьому випадку за замовчуванням до групи включаються всі файли, і SQL Server може самостійно планувати ефективний розподіл у межах бази даних.

У SQL Server реалізовані два режими аутентифікації: режим аутентифікації Windows (Windows Authentication Mode) і змішаний режим (Mixed).

Windows Authentication Mode – у SQL Server режим автентифікації за промовчанням. У цьому режимі для надання доступу користувачів або груп Windows до сервера баз даних SQL Server покладається виключно на аутентифікацію Windows. SQL Server підтверджує справжність користувачів майже таким самим способом як інші прикладні програми, використовуючи довірчі підключення. Коли використовується Windows Authentication Mode, адміністратор бази даних надає користувачам право підключатися до комп'ютера, на якому запущено SQL Server, надаючи їм право підключитися до SQL Server. Для відстеження входу до системи Windows використовуються ідентифікатори безпеки SID. Оскільки Windows використовується SID, адміністратор бази даних може надавати доступ для входу в систему безпосередньо користувачам або групам.

У змішаному режимі, достовірність користувачів може бути забезпечена Windows автентифікацією або власною автентифікацією SQL Server. Користувачі, завірені автентифікацією SQL Server, мають ім'я та пароль, які обслуговує сервер баз даних самостійно. Якщо SQL Server 2000 використовується в змішаному режимі, а клієнт і сервер можуть

використовувати для входу в систему протоколи автентифікації NTLM або Kerberos, при підтвердженні справжності користувачів сервер баз даних повністю покладається на Windows. Якщо клієнт не здатний використовувати стандартний вхід у систему Windows, SQL Server вимагає введення імені користувача і пароля, і порівнює їх з тим, що зберігає у своїх системних таблицях. Підключення, які покладаються на введення імені користувача та пароля, називаються не довіреними (non-trusted connections). Даний механізм успадкований від ранніх версії продукту і може бути видалений в наступних версіях. Використання його не рекомендується, оскільки імена облікових записів і паролі передаються через мережу практично незашифрованими. Власне, ім'я облікового запису передається відкритим текстом, пароль шифрується найпростішим алгоритмом на основі операції XOR. Змішаний режим залишено з двох причин: зворотна сумісність і для забезпечення можливості встановлення SQL Server на операційних системах Windows 98 або Windows Me.

SQL Server використовує дозволи для керування доступом у межах бази даних. Використовуючи привілеї (дозволи), можна визначити доступ до наступних об'єктів бази даних: таблиць, значень за умовчанням, правил, індексів, уявлень, тригерів, процедур, що зберігаються.

Не маючи відповідних дозволів, користувач може отримати доступ до SQL Server, але не до роботи з даними. Кожна база даних має незалежну систему дозволів. Таким чином, користувач, який має дозволи для роботи з однією БД, не зможе виконувати тих самих дій з іншого БД.

З точки зору системи доступу користувачів до об'єктів БД SQL Serv-er налічує 4 типи користувачів:

• системний адміністратор (sa);

• власник БД(dbo);

• власник об'єкта БД(dboo);

• Користувач БД.

Системний адміністратор знаходиться на вершині ієрархії, тому що має всі дозволи для доступу до об'єктів і виконання команд для будь-якої БД. Тільки він

може створювати БД і давати на це дозвіл іншим користувачам.

Користувач, який не є власником об'єкта, за замовчуванням не має жодних дозволів. Навіть при отриманні доступу до SQL Server він не зможе виконати жодних дій з даними. Однак будь-який користувач може зчитувати дані із системних таблиць, так як для цього не потрібно спеціальних дозволів.

Власник БД має всі права на виконання будь-яких дій з об'єктами своєї БД. Надалі власник може надати права доступу іншим користувачам або групам.

Користувач, який створив об'єкт БД, є його власником і автоматично отримує всі дозволи щодо роботи з ним. Інші користувачі, включаючи власника БД, не мають жодних дозволів для роботи з цим об'єктом, поки їх власник не надасть їх явно. Це не стосується системного адміністратора. Дозволи успадковуються під час створення об'єкта. Наприклад, якщо користувач отримав дозвіл на створення подання, він, створивши подання, стає його власником і може надавати або скасовувати дозволи на доступ до нього іншим користувачам БД.

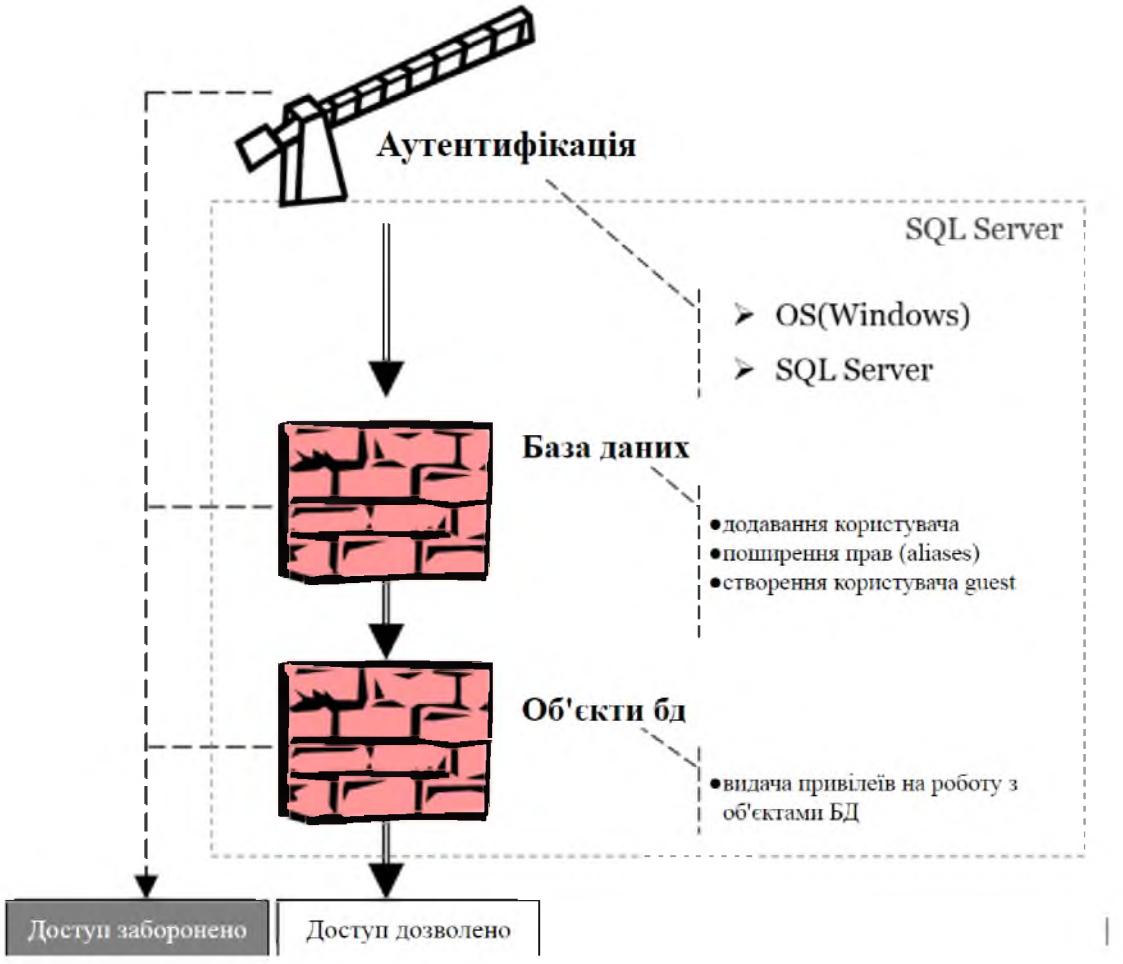

Рисунок 2.1 – Управління доступом до MS SQL Server

Використання ролей дуже схоже на використання груп Windows. Ролі дозволяють збирати користувачів у модулі, для яких можуть застосовуватися дозволи. Надання (granted), відхилення (denied) або скасування (revoked) дозволу для ролі також будуть дійсні для будь-якого члена цієї ролі.

Найпростіший шлях надання дозволів полягає у наданні чи забороні тих чи інших дій групам користувачів. При цьому важливо враховувати, що будь-який стандартний користувач входить у роль public(це стосується і користувача guest). SQL Server використовує останню версію дозволів. Наприклад, якщо якийсь дозвіл заборонено для користувача guest, а потім надано роль public, то користувач guest автоматично отримує цей дозвіл, оскільки входить у роль public.

SQL Server може вести аудит входу до системи сервера, який зберігається у Windows event log. Рівень аудиту можна вибрати в SQL Server Enterprise Manager або за допомогою розширеної збереженої процедури xp_loginconfig[6]. Можливі параметри налаштування аудиту:

• None – Не реєструється жодна інформація про аудит.

• Success – Реєструється лише успішний вхід до системи.

• Failure – Реєструється лише невдалий вхід до системи.

• All – Реєструється успішний та невдалий входи до системи.

Інформація аудиту зберігається у файли реєстрації помилок SQL Server.

Аудит засобами SQL Profiler

SQL Server Profiler дозволяє аналізувати широкий спектр внутрішніх подій SQL Server, включаючи події безпеки та аудиту.

Під час роботи SQL Profiler фіксує всі дії, що виконуються на SQL Server, а потім можна аналізувати ці дії. Зібрані дані можна переглядати на екрані в реальному часі, зберігати до текстового файлу або в таблиці SQL Server.

SQL Profiler дозволяє фіксувати фактично всі події, які мають місце в SQL Server, включаючи:

• дії кінцевих користувачів (всі SQL команди, вхід/вихід із системи,

використання ролей додатків);

• дії DBA (DDL, дії, відмінні від Grant/Revoke/Deny, а також події безпеки та конфігурування (БД або сервера));

• події безпеки (Grant/Revoke/Deny, додавання/видалення/зміна логіна користувача/ролі);

• сервісні події (резервування/відновлення/bulk insert/bcp/DBCC команди);

• події сервера (завершення, пауза, запуск);

• події аудиту (додавання, зміна, відключення аудиту).

2.6. Порівняльний аналіз системи безпеки розглянутих СУБД

Для того, щоб приступити до порівняння розглянутих СУБД, необхідно виробити критерії оцінювання, але в даному випадку слід обмовитися - якби якась одна з розглянутих СУБД була лідером за всіма параметрами, конкурентів просто не існувало б. Тому пропонується виконати порівняльний аналіз СУБД, але остаточний висновок у тому, яку СУБД використовувати, у кожному даному випадку залежатиме від поставлених завдань та умов:

1. Підтримка багатоплатформності. Те, що SQL Server розробляється тільки для платформи Windows, може розглядатися як його мінус, і як плюс. Мінус полягає в тому, що більшість серверних систем будуються на основі UNIX-систем (linux, freebsd, solaris) і SQL Server просто не може бути встановлений на даних серверах. Плюсом може служити його спеціальна інтеграція в Windows, він може використовувати всі переваги цієї операційної системи, починаючи від утиліт файлової системи до організації мережі доменів. Oracle та MySQL - багатоплатформні СУБД і можуть використовуватися на більшості сучасних серверних станцій. Цікавою та досить важливою особливістю MySQL є те, що він може поставлятися у вигляді вихідного коду, таким чином адміністратор може скомпілювати його «під себе» для подальшої роботи.

Переходячи до конкретних особливостей аналізованих СУБД, варто сказати, що в даний момент майже для кожної СУБД існує досить великий набір

зручного інструментарію та всіляких програмних надбудов, компонентів розширення. Наприклад, такі речі як автентифікація Kerberos або SSL-передача даних, вже давно по-своєму реалізовані в кожній окремій СУБД, і для кожної СУБД існує безліч утиліт резервного копіювання і подальшого відновлення при збоях.

2. Адміністрація. Адміністративні утиліти та інші інструменти для ранніх версій Oracle зазвичай вироблялися незалежними компаніями (випускалися як дорогі продукти, так і безкоштовні), а основні зусилля корпорації Oracle були зосереджені на розробці самої СУБД. Однак зараз ситуація суттєво змінилася, і в даний час Oracle постачає на ринок чимало різноманітних інструментів, серед яких крім засобів адміністрування є засоби розробки додатків, проектування даних, моделювання бізнес-процесів, сервери додатків, корпоративний портал. Втім, і в арсеналі Microsoft також є багато подібних продуктів, щоправда, не всі вони позиціонуються як такі. MySQL, в даному випадку, поступається в наочності та гнучкості Oracle і SQL Server - через те, що в СУБД використовується непроцедурний SQL-діалект, поки відсутня підтримка тригерів та засобів аудиту працездатності системи. З графічних інструментів адміністрування в MySQL є тільки mysqladmin, який дещо поступається за своїми функціональними можливостями аналогам від Oracle і Microsoft.

Говорячи про функціональні особливості безпеки даних, властивих тільки одній з СУБД, що розглядаються, слід згадати такі:

Oracle:

• VPD – Virtual Private Database, що дозволяє реалізувати мандатний контроль доступу для підприємства.

• Можливість контролювати розподіл процесорного часу сервера між активними клієнтами та лімітувати використання ресурсів сервера між користувачами.

• Кожна відкрита БД має свій робочий екземпляр Oracle – це робить СУБД Oracle більш стійкою до збоїв – збій в одній БД не торкається роботи інших відкритих БД.

SQL Server:

• Інтегрована з Active Directory та доменною структурою Windows – а це спрощує роботу з групами користувачів, надання їм ролей, контроль за доступом до інформації стає більш гнучким та зручним.

• Можливість відхиляти привілеї при роботі з об'єктами БД.

MySQL:

• Відсутні зумовлені ролі та облікові записи

• Контроль доступу базується на списках ACL і кардинально відрізняється від авторизації в Oracle та SQL Server

2.7. Стратегії виявлення та реагування на інциденти

Виявлення та реагування на інциденти в базах даних SQL є критично важливими, оскільки ці бази часто містять конфіденційну інформацію, включаючи особисті дані, фінансову інформацію, інтелектуальну власність. Швидке і ефективне реагування на інциденти допомагає захистити цю інформацію, запобігає фінансовим втратам, зберігає довіру клієнтів та позитивну репутацію організації. Також це важливо для відповідності нормативним вимогам, оскільки багато організацій зобов'язані дотримуватися певних стандартів захисту даних. Крім того, аналіз інцидентів дозволяє виявити та усунути слабкі місця в системах безпеки, що сприяє постійному вдосконаленню безпеки, а також запобігає повторенню подібних інцидентів у майбутньому.Управління інформаційною безпекою (УІБ) є критично важливим процесом, особливо у контексті баз даних SQL, які часто є ціллю кібератак. Цей процес включає збір та протоколювання даних про події, пов'язані з інформаційно-комунікаційними системами (ІКС), і використання цих даних для аналізу інцидентів, визначення збитків, та розробки стратегій для запобігання повторення інцидентів.

Координація та узгодженість дій є важливими в контексті управління інформаційною безпекою, особливо у базах даних SQL, оскільки вони допомагають забезпечити єдиний та ефективний підхід до захисту даних. Це

забезпечує, що усі члени команди, від IT-фахівців до керівництва, розуміють свої ролі та відповідальності у виявленні, реагуванні та запобіганні інцидентів безпеки. Без належної координації, зусилля можуть бути розпорошені, а відповіді на інциденти - неефективними. Для імплементації координації та узгодженості дій на підприємстві, важливо створити чіткі процедури та політики, провести навчання персоналу, регулярно перевіряти та оновлювати плани реагування на інциденти, а також стимулювати комунікацію та співпрацю між різними відділами. Також важливо встановити системи моніторингу та алертів, які дозволять оперативно реагувати на будь-які підозрілі дії або виявлені загрози. Все це сприяє створенню загальної безпекової культури на підприємстві, де кожен працівник усвідомлює свою роль у захисті цінних даних та інформаційних ресурсів.

Можливості збору та аналізу інформації є ключовими у рамках SQL безпеки, оскільки вони дозволяють організаціям ефективно виявляти, документувати та аналізувати потенційні загрози та вразливості. Збір даних, таких як журнали транзакцій, аудиту доступу, та системні логи, дозволяє отримати глибокий аналіз активності всередині баз даних. Це допомагає виявляти незвичайну або підозрілу активність, яка може свідчити про спроби несанкціонованого доступу або інші безпекові інциденти. Використання інструментів для аналізу цих даних може допомогти виявити шаблони, що вказують на потенційні проблеми безпеки, та сприяти вчасному втручанню. Для імплементації цих можливостей на підприємстві, потрібно встановити системи автоматичного збору та аналізу даних, такі як системи виявлення вторгнень або системи управління інформацією та подіями безпеки (SIEM). Також важливо забезпечити, щоб персонал мав необхідні навички та інструменти для інтерпретації зібраних даних. Регулярне оновлення цих систем і проведення тренінгів для персоналу є важливими для підтримки ефективності процесів збору та аналізу інформації. Такий підхід дозволяє не тільки реагувати на поточні загрози, але й прогнозувати майбутні ризики, забезпечуючи більш комплексний рівень безпеки для баз даних SQL.

Багатоджерельні сигнали про інциденти є важливими у рамках стратегій виявлення інцидентів SQL безпеки, оскільки вони забезпечують більш широкий та об'єктивний огляд безпекових подій. Полагодження на інформацію, що надходить з різноманітних джерел, дозволяє точніше ідентифікувати потенційні загрози та відслідковувати аномалії, які можуть бути пропущені при використанні одного джерела. Це може включати дані від систем виявлення вторгнень (IDS), систем управління інформацією та подіями безпеки (SIEM), аудиту доступу, журналів транзакцій, та повідомлень від користувачів. Використання цих різних джерел допомагає виявити складні атаки, які можуть бути неочевидними при розгляді лише одного типу даних. Для імплементації багатоджерельних сигналів про інциденти на підприємстві, необхідно інтегрувати різні системи моніторингу та збору даних, забезпечити їх належне налаштування та синхронізацію. Також важливо провести навчання персоналу для вмілого користування цими системами та розуміння, як інтерпретувати дані з різних джерел. Регулярне оновлення та оцінка ефективності цих систем дозволить підтримувати високий рівень безпеки та гнучко реагувати на змінні загрози в сфері SQL безпеки. Аналіз Повторюваних Подій: Часте повторення певних підозрілих подій, зафіксованих IDS, може свідчити про потенційний інцидент.

Використання журналів SQL може значно допомогти у рамках стратегій виявлення інцидентів SQL безпеки. Ці журнали забезпечують детальний запис усіх транзакцій та змін, що відбуваються в базі даних, включаючи запити та зміни в даних. Це дозволяє адміністраторам систем та фахівцям з безпеки відстежувати весь потік даних та операцій, що здійснюються в системі. Журнали допомагають ідентифікувати незвичайні або підозрілі дії, такі як несанкціоновані запити до даних, незвичайні зміни у даних, або спроби доступу до конфіденційної інформації. Вони також є важливим інструментом при проведенні форензичного аналізу після виявлення інцидентів, дозволяючи визначити, які дані були вплинуті та яким чином. Для ефективного використання журналів у рамках стратегій безпеки, необхідно забезпечити, щоб системи

журналювання були правильно налаштовані та захищені від несанкціонованого доступу. Також важливо регулярно перевіряти та аналізувати ці журнали, використовуючи автоматизовані інструменти та системи аналізу даних, щоб виявляти потенційні проблеми безпеки на ранніх стадіях. Це дозволяє підприємствам своєчасно реагувати на інциденти та запобігати можливим збиткам, пов'язаним з втратою даних або порушенням безпеки.

Стратегії Реагування на Інциденти

Впровадження плану реагування на інциденти є ключовим елементом стратегії безпеки баз даних SQL, оскільки воно забезпечує організований та ефективний спосіб реагування на безпекові порушення. Такий план допомагає визначити конкретні кроки, які необхідно вжити при виявленні інциденту, що може включати ізоляцію вразливих систем, проведення форензичного аналізу для виявлення джерела та обсягу інциденту, та відновлення нормальної роботи системи. Наявність чітко визначеного плану дозволяє швидко мобілізувати ресурси та персонал, необхідний для реагування на інцидент, зменшуючи можливий збиток та скорочуючи час відновлення. План також повинен включати процедури повідомлення керівництва та зовнішніх органів, якщо це потрібно, та визначати критерії для оцінки важливості та впливу інциденту. Регулярні тренування та вправи допомагають забезпечити, що персонал готовий діяти відповідно до плану, а періодичний перегляд та оновлення плану забезпечують, що він залишається актуальним у світлі нових загроз та змін у технологічному ландшафті. В цілому, впровадження плану реагування на інциденти є важливим для забезпечення того, що організація може ефективно управляти та мінімізувати вплив інцидентів безпеки, забезпечуючи захист важливих даних та ресурсів. УІБ в контексті баз даних SQL вимагає комплексного підходу, що поєднує технологічні рішення та чітку організаційну структуру. З використанням сучасних систем управління інцидентами інформаційної безпеки (СУІБ), можна значно підвищити ефективність цих процесів, інтегруючи апаратно-програмне забезпечення, людські ресурси та інформаційну інфраструктуру організації в єдину систему.

2.8. Методи покращення безпеки СУБД

Підвищення безпеки систем управління базами даних (СУБД) є невід'ємною частиною сучасних стратегій захисту інформації, особливо в умовах, коли кіберзагрози стають дедалі складнішими та різноманітнішими. Оскільки бази даних часто містять критичні корпоративні дані, їх захист від несанкціонованого доступу, злому або втрати даних є важливим для забезпечення неперервності бізнес-процесів та збереження конфіденційності інформації.

Шифрування даних є одним з найефективніших методів посилення безпеки в системах управління базами даних SQL, оскільки воно дозволяє забезпечити конфіденційність інформації навіть у випадку несанкціонованого доступу. Шифруванням можна захистити як дані, що зберігаються (at-rest), так і дані, що передаються (in-transit). Шифруючи дані в базі даних, можна гарантувати, що навіть якщо зловмисники отримають доступ до бази даних, вони не зможуть прочитати або використовувати конфіденційну інформацію без відповідного ключа шифрування. Це особливо важливо для захисту чутливих даних, таких як особиста інформація користувачів, фінансові дані та інтелектуальна власність. Шифрування даних, що передаються, також запобігає можливості перехоплення даних під час їх трансферу між сервером і клієнтами.

Управління доступом та аутентифікація користувачів є фундаментальними аспектами безпеки систем управління базами даних SQL і відіграють ключову роль у покращенні загального рівня безпеки СУБД. Це полягає у встановленні строгих процедур і механізмів для визначення того, хто може отримувати доступ до бази даних і які саме операції ці користувачі можуть виконувати. Аутентифікація забезпечує впевненість у тому, що особа, яка намагається отримати доступ до системи, дійсно є тим, за кого себе видає, шляхом використання паролів, біометричних даних або інших методів аутентифікації. Після успішної аутентифікації, управління доступом визначає, до яких частин

бази даних користувач має доступ та які операції він може виконувати. Це включає в себе встановлення різних рівнів дозволів для різних користувачів або груп, обмеження доступу до чутливих даних та контроль за виконанням запитів до бази даних. Ефективне управління доступом допомагає запобігти несанкціонованому доступу, знижуючи ризик витоку або пошкодження даних, а також допомагає гарантувати, що користувачі мають доступ лише до тих даних, які необхідні для їх роботи. Регулярне переглядання та оновлення політик доступу та процедур аутентифікації є критично важливими для підтримки високого рівня безпеки, особливо у світлі постійного розвитку кіберзагроз та змін в організаційній структурі.

Моніторинг та аудит є важливими компонентами стратегії забезпечення безпеки систем управління базами даних SQL, оскільки вони дозволяють відслідковувати, записувати та аналізувати всі активності, що відбуваються в базі даних. Це включає в себе моніторинг запитів до бази даних, зміни даних, а також спроби доступу до бази даних. Завдяки цьому, можна виявляти незвичайну або підозрілу активність, що може вказувати на спроби несанкціонованого доступу або інші безпекові порушення. Аудит баз даних забезпечує детальний запис про виконані операції, що дозволяє провести форензичний аналіз у випадку виявлення інцидентів безпеки. Це допомагає зрозуміти, як інцидент стався, які дані були залучені та хто був відповідальний за порушення. Регулярний моніторинг та аудит також сприяють підвищенню загальної прозорості та керованості системи, дозволяючи своєчасно виявляти та виправляти слабкі місця у безпеці. Для ефективної реалізації цих процесів, необхідно використовувати спеціалізоване програмне забезпечення для моніторингу та аудиту, налаштовувати відповідні правила та сповіщення для виявлення підозрілих дій, а також регулярно переглядати та аналізувати зібрані дані. Це дозволяє не лише реагувати на поточні загрози, але й прогнозувати майбутні ризики, що є ключовим для підтримання надійної безпеки в базах даних SQL.

Використання машинного навчання для виявлення аномалій може значно

посилити безпеку систем управління базами даних SQL, оскільки це дозволяє автоматично ідентифікувати потенційно шкідливу або незвичайну активність, яка може вказувати на кібератаки або внутрішні порушення безпеки. Особливо алгоритми машинного навчання ефективні щоб виявити відхилення від звичайних моделей поведінки, такі як несподівані запити до бази даних, зміни в обсягах даних або нетипові шаблони доступу. Це допомагає виявляти зломи або витоки даних на ранніх стадіях, що значно знижує ризики та можливі збитки. Для успішної імплементації також важливо регулярно оновлювати та налагоджувати модель, забезпечуючи, що вона відповідає змінам в поведінці користувачів та новим загрозам. Крім того, команда безпеки має бути навчена правильно інтерпретувати сповіщення від системи машинного навчання, щоб швидко та ефективно реагувати на можливі інциденти. Цей підхід дозволяє значно підвищити рівень безпеки СУБД, роблячи захист більш адаптивним і прогнозованим.

Використання технології блокчейну для аудиту може значно посилити безпеку систем управління базами даних SQL, оскільки блокчейн пропонує високий рівень прозорості, незмінності та відстежуваності даних. У контексті аудиту, блокчейн може використовуватись для створення незмінних та повністю прозорих журналів всіх транзакцій та змін у базі даних. Кожна операція записується в блокчейні як окремий блок, який містить відмітку часу та є пов'язаним з попередніми блоками, створюючи ланцюг, який неможливо змінити без виявлення. Ця особливість блокчейну дозволяє створити систему аудиту, де записи є повністю незмінними та легко перевіряються, забезпечуючи відмінну прозорість та довіру до даних. Такий підхід є корисним при розслідуванні інцидентів безпеки, оскільки він дозволяє точно відстежити, коли і які дані були змінені, та встановити відповідальних осіб. Щоб імплементувати блокчейн для аудиту в СУБД SQL, необхідно інтегрувати блокчейн з існуючою системою бази даних. Це може вимагати розробки або впровадження спеціальних інструментів або платформ, які можуть автоматично записувати важливі транзакції в блокчейн. Важливо також врахувати питання масштабування та ефективності,

оскільки запис кожної транзакції в блокчейн може вимагати додаткових ресурсів. В цілому, інтеграція блокчейну для аудиту в СУБД SQL може значно посилити захист даних, забезпечуючи незмінність історії транзакцій та підвищуючи рівень довіри до системи аудиту.

Розподілена обробка даних може посилити безпеку систем управління базами даних SQL, оскільки цей підхід дозволяє розподілити дані та запити до бази даних між кількома вузлами або серверами, замість зосередження всієї інформації в одному місці. Це робить систему менш вразливою до атак, оскільки навіть у випадку компрометації одного вузла, інші частини системи залишаються недоторканими та продовжують функціонувати нормально. Крім того, розподілена обробка даних дозволяє збалансувати навантаження та забезпечити більшу відмовостійкість, оскільки навіть у випадку відмови одного сервера, інші можуть продовжувати роботу без перерв. Для імплементації розподіленої обробки даних в системах SQL, необхідно створити архітектуру, яка дозволяє розподіляти дані між різними вузлами або серверами. Це може включати в себе використання технологій кластеризації, реплікації даних та балансування навантаження. Також важливо забезпечити належне шифрування та безпеку передачі даних між вузлами, а також ефективне управління доступом та аутентифікацію користувачів у розподіленій системі. Окрім цього, необхідно встановити регулярні процедури моніторингу та оновлення всіх компонентів системи для забезпечення їх безпеки та ефективності. Розподілена обробка даних як стратегія безпеки дозволяє створити більш гнучку та відмовостійку систему, підвищуючи загальний рівень захисту в базах даних SQL.

Застосування контейнеризації в системах управління базами даних SQL може значно підсилити безпеку, оскільки ця технологія дозволяє ізолювати додатки та їх залежності в окремих контейнерах. Ця ізоляція запобігає сценаріям, в яких потенційні уразливості або зловмисні дії в одному додатку можуть поширюватися на інші додатки або на головну систему. Контейнеризація дозволяє також легко оновлювати та впроваджувати патчі безпеки, забезпечуючи, що всі компоненти системи завжди оновлені та захищені. У

контексті SQL баз даних, контейнеризація може бути використана для запуску інстанцій бази даних у відокремлених середовищах, забезпечуючи додатковий рівень безпеки. Це також полегшує управління конфігурацією та розгортання, оскільки кожен контейнер може мати свою власну конфігурацію та залежності, і не впливатиме на інші частини системи. Для імплементації контейнеризації в СУБД SQL, можна використовувати інструменти як Docker або Kubernetes, які дозволяють створювати, розгортати та керувати контейнерами. Важливо також забезпечити належну конфігурацію мережі та безпеки для контейнерів, включаючи налаштування мережевих правил, що обмежують небажаний трафік, та імплементацію політик безпеки на рівні контейнерів. Контейнеризація, як стратегія безпеки, допомагає створити більш гнучку, модульну та безпечну інфраструктуру для СУБД SQL, значно підвищуючи рівень захисту від зовнішніх та внутрішніх загроз.

2.9. Висновки

Отже, протягом цього розділу ми розглянули різноманітні методи та підходи, які можуть бути використані для покращення безпеки систем управління базами даних SQL. Специфічних особливості безпеки для конкретних систем управління базами даних, таких як MySQL, Oracle та MS SQL Server, до загальних стратегій безпеки СУБД, які ми вже обговорили, може значно підсилити захист даних. Кожна з цих СУБД має свої унікальні функції та вразливості, які потребують спеціалізованих підходів до безпеки.

Для кожної з цих СУБД, ефективна стратегія безпеки повинна включати комбінацію загальних методів безпеки, таких як контейнеризація, моніторинг, машинне навчання та специфічні для платформи налаштування та практики. Це вимагає ретельного планування, постійного оновлення та адаптації до нових викликів у галузі кібербезпеки. Важливо також проводити регулярні огляди безпеки та аудити, щоб забезпечити, що всі заходи безпеки залишаються актуальними та ефективними.

Кожен з цих підходів пропонує унікальні переваги та сприяє підвищенню

загальної безпеки СУБД SQL. Наприклад, шифрування даних забезпечує захист інформації навіть у випадку її витоку, управління доступом та аутентифікація допомагають контролювати, хто має доступ до системи, а моніторинг та аудит дозволяють відстежувати всі дії в системі. Машинне навчання використовується для автоматизації процесу виявлення аномалій, тоді як блокчейн вносить незмінність та прозорість в процеси аудиту. Окрім цього, розподілена обробка даних та контейнеризація додатково підвищують відмовостійкість та гнучкість системи.

Важливо розуміти, що жоден із цих методів сам по собі не може забезпечити повну безпеку, але їх комбінація та інтеграція в комплексну безпекову стратегію можуть значно знизити ризики та підвищити здатність системи протистояти потенційним загрозам. Впровадження цих методів вимагає глибокого розуміння конкретних потреб та загроз організації, а також регулярного оновлення та адаптації до нових викликів у сфері кібербезпеки.

РОЗДІЛ 3. ЕКОНОМІЧНИЙ РОЗДІЛ

Сума витрат на забезпечення безпеки не повинна перевищувати цінність даних, що охороняються, тобто. збитків підприємства у разі втрати таємності, цілісності чи доступності конфіденційної інформації. Тому при виборі СУБД слід пам'ятати, що MySQL – продукт, що вільно розповсюджується, на відміну від ORACLE і SQL Server. MySQL – це продукт діяльності ентузіастів, які живуть у всьому світі. Тому він безкоштовний, але це зовсім не говорить про те, що він малонадійний чи недостатньо конкурентоспроможний. Погляди Oracle і Microsoft на шляху зниження повної вартості володіння їх СУБД істотно різняться. Oracle розглядає це питання з позиції надійності продукту, тоді як Microsoft – з позиції цін на окремі продукти, додаткові утиліти, послуги та послуги. Ціна на продукти Oracle зазвичай набагато перевищує ціни на аналогічні продукти Microsoft, особливо з урахуванням того, що у складі Microsoft SQL Server є утиліти та сервіси, які при виборі Oracle слід придбати окремо (і, зрозуміло, за додаткову плату). Вважається, що надійність СУБД Oracle може бути вищою, ніж надійність SQL Server, особливо при експлуатації Oracle під керуванням операційної системи з високою надійністю та стійкістю до відмови. На практиці обчислити повну вартість володіння тим та іншим продуктом можна, тільки маючи відомості про те, що є конкретним підприємством, у чому полягає його діяльність, і яка його інфраструктура, тобто необхідно всебічно розглянути використання продукту в контексті проекту, в якому він буде застосовуватись.

Вимоги до ресурсної частини найменші у MySQL, більш вимогливий MS SQL Server, а Oracle здається і натомість цих СУБД справжнім «танком», переважно через те, що кожної відкритої БД у пам'яті створюється його новий екземпляр.

3.1. Розрахунок капітальних витрат

Капітальні інвестиції:

- вартість розробки проекту інформаційної безпеки (розробка схем пристроїв, політики функціонування системи тощо);
- вартість створення основного й додаткового програмного забезпечення (ПЗ);

- витрати на первісні закупівлі апаратного забезпечення;
- витрати на навчання технічних фахівців і обслуговуючого персоналу.

Спершу розрахуємо час, який буде витрачено на створення ПЗ:

$$t = tmз + tв + ta + tпр + tопр + tд, \text{ годин,} \quad (3.1)$$

де $tmз$ – тривалість складання технічного завдання на розробку ПЗ;

$tв$ – тривалість вивчення ТЗ, літературних джерел за темою тощо;

ta – тривалість розробки блок-схеми алгоритму;

$tпр$ – тривалість програмування за готовою блок-схемою;

$tопр$ – тривалість опрацювання програми на ПК;

$tд$ – тривалість підготовки технічної документації на ПЗ.

Умовна кількість оперантів у програмі:

$$Q = q \cdot c\,(1 + p), \text{ штук,} \quad (3.2)$$

де q – очікувана кількість операторів – 21;

c – коефіцієнт складності програми – 1.3;

p – коефіцієнт корекції програми в процесі її опрацювання – 0.05.

$$Q = 21 \cdot 1.3(1 + 0.05)=28.6, \text{ штук.}$$

Оцінка тривалості складання технічного завдання на розробку ПЗ **ттз** – 3 год. Тривалість вивчення технічного завдання:

$$tв = \frac{Q \cdot B}{(75...85) \cdot k} = \frac{28.6 \cdot 1.2}{80 \cdot 1} = 0.429 \text{, годин.} \quad (3.3)$$

де B – коефіцієнт збільшення тривалості етапу внаслідок недостатнього

опису завдання, В = 1,2...1,5; k – коефіцієнт, що враховує кваліфікацію програміста і визначається стажем роботи за фахом: від 2 до 3 років – 1,0;

Тривалість розробки блок-схеми алгоритму:

$$ta = \frac{Q}{(20...25)\cdot k} = \frac{28.6}{20\cdot 1} = 1.43 \text{ , годин.} \quad (3.4)$$

Тривалість складання програми за готовою блок-схемою:

$$tпр = \frac{Q}{(20...25)\cdot k} = \frac{28.6}{20\cdot 1} = 1.43 \text{ , годин.} \quad (3.5)$$

Тривалість опрацювання програми на ПК:

$$tопр = \frac{1,5Q}{(4...5)\cdot k} = \frac{1.5\cdot 28.6}{4\cdot 1} = 10.725 \text{ , годин.} \quad (3.6)$$

Тривалість підготовки технічної документації на ПЗ:

$$t_д = \frac{Q}{(15...20)\cdot k} + \frac{Q}{(15...20)}\cdot 0,75 = \frac{28.6}{15\cdot 1} + \frac{28.6}{15}\cdot 0.75 = 2.87 \text{ , годин.} \quad (3.7)$$

$$t = 3 + 0.429 + 1.43 + 1.43 + 10.725 + 2.87 = 19.884 \text{ годин.}$$

Розрахунок витрат на створення програмного продукту

$$К_{пз} = З_{зп} + З_{мч}. \text{ грн} \quad (3.8)$$

Заробітна плата виконавця враховує основну і додаткову заробітну плату, а також відрахування на соціальне потреби (пенсійне страхування, страхування на випадок безробіття, соціальне страхування тощо) и визначається за формулою:

$$З_{зп} = t \cdot З_{пр} = 19.884 \cdot 309.52 = 6154.50 \text{ , грн,} \qquad (3.9)$$

де t – загальна тривалість створення ПЗ, годин;

Зпр – средньогодинна заробітна плата програміста з нарахуваннями, грн/годину.

$$З_{пр} = \frac{З_м}{168} = 309.52 \text{ , грн/годину.} \qquad (3.10)$$

де $З_м$ – середня заробітна плата на місяць – 52000 грн.

Вартість машинного часу для налагодження програми на ПК визначається за формулою:

$$З_{мч} = t_{опр} \cdot С_{мч} + t_д = 10.725 \cdot 1.2 + 2.87 = 15.74 \text{ , грн.} \qquad (3.11)$$

де $t_{опр}$ – трудомісткість налагодження програми на ПК, годин;

$t_д$ – трудомісткість підготовки документації на ПК, годин;

$С_{мч}$ – вартість 1 години машинного часу ПК, грн./година.

Вартість 1 години машинного часу ПК визначається за формулою:

$$С_{мч} = P \cdot t_{нал} \cdot С_е + \frac{Ф_{зал} \cdot Н_а}{F_р} + \frac{К_{лпз} \cdot Н_{апз}}{F_р}$$

$$С_{мч} = 0.5 \cdot 2.1 + \frac{3000 \cdot 0.1}{1920} = 1.2 \text{ , грн/год.} \qquad (3.12)$$

де P – встановлена потужність ПК, 0.5 кВт;

$С_е$ – тариф на електричну енергію, 2.1 грн/кВт·година;

$Ф_{зал}$ – залишкова вартість ПК на поточний рік, 3000 грн.;

$Н_а$ – річна норма амортизації на ПК, 0.1 частки одиниці;

$Н_{апз}$ – річна норма амортизації на ліцензійне програмне забезпечення, частки одиниці;

$К_{лпз}$ – вартість ліцензійного програмного забезпечення, грн.;

$F_р$ – річний фонд робочого часу (за 40-годинного робочого тижня $F_р$ = 1920 год).

$$К_{пз} = 6154.50 + 15.74 = 6170.24 \text{ грн.} \quad (3.8)$$

Таким чином, капітальні (фіксовані) витрати на проектування та впровадження проектного варіанта системи інформаційної безпеки складають:

$$К = К_{пз} + К_{аз} + К_{навч} + К_{н}, \text{ тис. грн.} \quad (3.13)$$

де $К_{пз}$ – вартість створення програмного продукту, тис. грн;

$К_{аз}$ – вартість закупівлі апаратного забезпечення та допоміжних матеріалів, тис. грн;

$К_{навч}$ – витрати на навчання технічних фахівців і обслуговуючого персоналу, тис. грн;

$К_{н}$ – витрати на встановлення обладнання та налагодження системи інформаційної безпеки, тис. грн.

Таблиця 3.1 Вартість закупівлі апаратного забезпечення та допоміжних матеріалів

Назва	Вартість, грн.
Сервер: BAS-Dedicated4 12x3.8 GHz CPU 1000 GB NVMe-диск	79920
Ліцензія на використання: Microsoft SQL Server Standard edition	36593
SIEM: Logpoint	65000
Разом	181513

Каз = 181.513 тис. грн

Витрати на навчання технічних фахівців і обслуговуючого персоналу, це є підготовчі курси з адміністрування та обслуговування системи виявлення

вторгнень що складають 5.5 тис. грн;

$К_{навч}$ =5.5 тис. грн.

Витрати на встановлення обладнання та налагодження системи інформаційної безпеки складають, 8 тис. грн.

$К_н$ = 8 тис. грн.

$$К = 6170 + 181513 + 5500 + 8000 = 201183 \text{ грн.} \quad (3.13)$$

3.2. Експлуатаційні витрати**:**

$$С_к = С_н + С_а + С_з + С_{єв} + С_е + С_{ел} + С_{тос} \quad (3.14)$$

де витрати на навчання адміністративного персоналу й кінцевих користувачів($С_н$). визначаються за даними організації з проведення тренінгів персоналу, курсів підвищення кваліфікації – 5.5 тис. грн.

Річний фонд амортизаційних відрахувань ($С_а$) визначається у відсотках від суми капітальних інвестицій за видами основних фондів і нематеріальних активів (ПЗ) – 20% або 40236,60 грн.

Річний фонд заробітної плати інженерно-технічного персоналу, що обслуговує систему інформаційної безпеки ($С_з$), складає:

$$С_з = З_{осн} + З_{дод} = 6700 \cdot 12 + 6700 \cdot 0.22 \cdot 12 = 98\,088 \text{ грн.} \quad (3.15)$$

де $З_{осн}$, $З_{дод}$ – основна мінімальна заробітна плата на 01.01.2023, грн на рік.

Единий соціальний внесок – 0.22, частки одиниці;

Вартість електроенергії, що споживається апаратурою системою інформаційної безпеки протягом року ($С_е$), визначається за формулою:

$$С_{ел} = Р \cdot F_р \cdot Ц_е = 0.5 \cdot 365 \cdot 24 \cdot 2.1 = 9\ 198 \text{ грн}, \quad (3.16)$$

де Р – встановлена потужність апаратури інформаційної безпеки, кВт;

$F_р$ – річний фонд робочого часу системи інформаційної безпеки (визначається виходячи з режиму роботи системи інформаційної безпеки);

$Ц_е$ – тариф на електроенергію, грн/кВт·годин

3.3. Оцінка можливого збитку від атаки (злому) на вузол або сегмент корпоративної мережі

Кінцевим результатом впровадження й проведення заходів щодо забезпечення інформаційної безпеки є величина відвернених втрат, що розраховується, виходячи з імовірності виникнення інциденту інформаційної безпеки й можливих економічних втрат від нього. По суті, ця величина відображає ту частину прибутку, що могла бути втрачена.

Загалом можливо виділити такі види збитку, що можуть вплинути на ефективність комп'ютерної системи інформаційної безпеки (КСІБ):

- порушення конфіденційності ресурсів КСІБ (тобто неможливість доступу до них неавторизованих суб'єктів або несанкціонованого використання каналів зв'язку);
- порушення доступності ресурсів КСІБ (тобто можливість доступу до них авторизованих суб'єктів (завжди, коли їм це потрібно);
- порушення цілісності ресурсів КСІБ (тобто їхня неушкодженість);
- порушення автентичності ресурсів КСІБ (тобто їхньої дійсності, непідробленості).

Вихідні дані:

$t_п$ =48 годин – час простою вузла або сегмента корпоративної мережі внаслідок атаки, годин;

$t_в$=24 годин – час відновлення після атаки персоналом, що обслуговує корпоративну мережу, годин;

$t_{ви}$=12 годин – час повторного введення загубленої інформації співробітниками атакованого вузла або сегмента корпоративної мережі, годин;

$З_о$= 6700 грн – місячна заробітна плата обслуговуючого персоналу (адміністраторів та ін.) з нарахуванням єдиного соціального внеску, грн на місяць;

$З_с$= 8174 грн – місячна заробітна плата співробітника атакованого вузла або сегмента корпоративної мережі з нарахуванням єдиного соціального внеску, грн на місяць;

$Ч_о$=4 – чисельність обслуговуючого персоналу (адміністраторів та ін.), осіб.;

$Ч_с$=10 – чисельність співробітників атакованого вузла або сегмента корпоративної мережі, осіб.;

О = 150 000 грн – обсяг чистого прибутку/дохід від реалізації/ атакованого вузла або сегмента корпоративної мережі, грн у рік, або оподаткований прибуток атакованого вузла або сегмента корпоративної мережі;

$П_{зч}$ = 1700 грн – вартість заміни встаткування або запасних частин, грн;

I=1 – число атакованих вузлів або сегментів корпоративної мережі;

N = 16 – середнє число можливих атак на рік.

Упущена вигода від простою атакованого вузла або сегмента корпоративної мережі становить:

$$U = П_п + П_в + V, \text{ грн.} \quad (3.15)$$

де $П_п$ – оплачувані втрати робочого часу та простої співробітників атакованого вузла або сегмента корпоративної мережі, грн;

$П_в$ – вартість відновлення працездатності вузла або сегмента корпоративної мережі (переустановлення системи, зміна конфігурації та ін.), грн;

U – втрати від зниження обсягу продажів за час простою атакованого вузла або сегмента корпоративної мережі, грн.

Втрати від зниження продуктивності 3 співробітників з ЗП атакованого вузла або сегмента корпоративної мережі являють собою втрати їхньої заробітної плати (оплата непродуктивної праці) за 60 годин простою внаслідок атаки:

$$П_п = \frac{\sum Зс \cdot Чс}{} \cdot t_п, \qquad (3.16)$$

де F – місячний фонд робочого часу (при 40-а годинному робочому тижні становить 160-176 ч).

$$П_п = \frac{\sum 6700 \cdot 10}{160} \cdot 48 = 20100 \text{ , грн.}$$

Витрати на відновлення працездатності вузла або сегмента корпоративної мережі включають кілька складових:

$$П_в = П_{ви} + П_{пв} + П_{зч}, \text{ грн.} \qquad (3.17)$$

де $П_{ви}$ – витрати на повторне уведення інформації, грн;

$П_{пв}$ – витрати на відновлення вузла або сегмента корпоративної мережі, грн;

$П_{зч}$ – вартість заміни устаткування або запасних частин, грн.

Витрати на повторне введення інформації $П_{ви}$ розраховуються виходячи з розміру заробітної плати 6700 грн 3 співробітників атакованого вузла або сегмента корпоративної мережі $З_с$, які зайняті повторним введенням втраченої інформації, з урахуванням необхідного для цього часу $t_{ви}$=12:

$$П_{ви} = \frac{\sum 6700}{160} \cdot 12 = 502.5 \text{ , грн.} \qquad (3.18)$$

Витрати на відновлення вузла або сегмента корпоративної мережі $П_{пв}$ визначаються часом відновлення після атаки $t_в = 24$ і розміром середньогодинної заробітної плати обслуговуючого персоналу (адміністраторів):

$$П_{пв} = \frac{\sum 6700}{160} \cdot 24 = 1005 \text{ , грн.} \quad (3.19)$$

$$П_в = 502.5 + 1005 + 1760 = 3267.5 \text{ грн.} \quad (3.17)$$

Втрати від зниження очікуваного обсягу продаж в 200 000 грн за 90 годин простою атакованого вузла або сегмента корпоративної мережі виходячи із середньогодинного обсягу продажів і сумарного часу простою атакованого вузла або сегмента корпоративної мережі:

$$V = \frac{O}{F_2} \cdot (t_п + t_в + t_{від}) = \frac{150000}{9340} \cdot (48 + 24 + 12) = 1349.04 \text{ , грн} \quad (3.20)$$

де $F_г$ – річний фонд часу роботи організації (прийом заказів інтернетмагазином) становить близько 9340 ч.

$$U = П_п + П_в + V = 20100 + 3267.5 + 1349.04 = 24716{,}54 \text{ грн.} \quad (3.15)$$

Таким чином, загальний збиток від атаки на вузол або сегмент корпоративної мережі організації складе

$$B = \sum_i \sum_n U = 24716{,}54 \cdot 16 \cdot 1 = 395464{,}64 \text{ грн.} \quad (3.21)$$

3.4. Загальний ефект від впровадження системи інформаційної безпеки

Загальний ефект від впровадження системи інформаційної безпеки визначається з урахуванням ризиків порушення інформаційної безпеки і становить:

$$E = B \cdot R - C = 395464.64 \cdot 0.7 - 98088 = 178737.24 \text{ грн,} \quad (3.22)$$

де В – загальний збиток від атаки на вузол або сегмент корпоративної мережі, грн;

R – очікувана імовірність атаки на вузол або сегмент корпоративної мережі, частки одиниці;

С – щорічні витрати на експлуатацію системи інформаційної безпеки, тис. грн.

3.5.Визначення та аналіз показників економічної ефективності системи інформаційної безпеки

Коефіцієнт повернення інвестицій ROSI:

$$ROSI = \frac{E}{K} = \frac{178.74}{6.85} = 26.09 \text{ ,частки одиниці} \quad (3.23)$$

де Е – загальний ефект від впровадження системи інформаційної безпеки, грн;

К – капітальні інвестиції за варіантами, що забезпечили цей ефект, грн.

Термін окупності:

$$T_o = \frac{K}{E} = \frac{1}{ROSI} = 0.04 \text{ років} \quad (3.23)$$

3.6. Висновки

В економічному розділі в результаті обчислюваних витрат на впровадження методів захисту бази даних SQL, було підтверджено його економічну ефективність та термін окупності витрат. Проект економічно доцільний та його можно використовувати на підприємстві.

ВИСНОВКИ

У цій кваліфікаційній роботі було всебічно розглянуто важливість інформаційної безпеки в SQL базах даних через аналіз заходів захисту та особливостей безпеки в системах MySQL, Oracle та MS SQL Server. Було визначено, що ефективне впровадження та підтримка заходів безпеки є економічно виправданими, враховуючи потенційні втрати від інцидентів пов'язаних з безпекою даних. Рекомендації, надані в роботі, мають на меті допомогти організаціям у підвищенні рівня захищеності їхніх баз даних, одночасно оптимізуючи витрати, пов'язані з управлінням цих систем. Робота не тільки поглиблює розуміння проблем безпеки бази даних SQL, але й пропонує практичні рішення, які відповідають галузевим стандартам і вирішують поточні загрози кібербезпеці. Особливістю даної роботи є те, що вона допомагає сформувати основні поняття про представлені на даний момент методи захисту даних у СУБД і може стати зручним помічником для прийняття власного рішення та власних висновків. Крім того, дана робота вносить вагомий вклад у розуміння ролі користувацької свідомості та відповідальності у підтримці інформаційної безпеки. Через аналіз різних аспектів безпеки баз даних, робота підкреслює необхідність комплексного підходу, що поєднує технічні, адміністративні та освітні стратегії для ефективного захисту даних. Висновки цієї роботи можуть слугувати основою для розробки більш ефективних політик та процедур у сфері інформаційної безпеки в організаціях.

ПЕРЕЛІК ПОСИЛАНЬ

[1]. Bobrowski S. – *«Посібник з концепцій сервера. Oracle 21c»*, Переклад: Compek Systems, Київ, 2016.

[2]. Delaney K. - *"Inside Microsoft SQL Server 2008"*, Microsoft Press., 2008.

[3]. DuBois P. - *"MySQL. Second Edition"*, Sams, 2015.

[4]. Suehring S. – *"MySQL Bible"*, Wiley Publishing, Inc., 2002.

[5]. Waymire R., Thomas B. – *"Безпека Microsoft SQL Server 2019"*, (http://www.sql.ru/articles/mssql/03052501MSSQL2KSecurity.shtml)

[6]. Аткінсон Л. - *«Бібліотека професіонала MySQL»*, М.: Вильямс, 2014.

[7]. Гарсіа-Моліна Гектор, Джеффрі Д.Ульман, Дженніфер Уідом - *«Системи баз данных. Повний курс»*, М.: Вільямс, 2013.

[8]. Kalen Delaney. – «*Microsoft SQL Server 2012»*, Sams, 2013.

[9]. Елманова Н., Федоров А. – *«Oracle и Microsoft SQL Server: минуле, теперішне та майбутнє»*, журнал «Комп'ютерПресс», 2017.

[10]. Кайт Т. – *«Детальний контроль доступу та контексти програми»*, (http://www.interface.ru/oracle/kontrol_1.htm)

[11]. Кайт Т. - *«Oracle для професіоналів»*, М.: «ДиаСофт», 2013.

[12]. Козленко Лилия – *«Інформаційна безпека у сучасних системах управління базами даних»*, Комп'ютерПресс №3, 2020.

[13]. Конноли Т., Бегг К. – «*Бази даних: проектування, реалізація та супроводження. Теорія та практика»* 3-е видання.: Пер. з англ. - М.: Вільямс, 2017.

[14]. Маллинс С. К. - *«Адміністрування баз данных»*, М.: Кудиц-Образ, 2018.

[15]. Покровский П., Четвертнев И. – *«Методы аутентифікації в ORACLE»*, Журнал "Windows & .NET Magazine/RE", 2019.

[16]. Рейли Майкл Д. – *«Основи системи безпеки SQL Server»*, Журнал «Windows & .NET Magazine/RE», 2017.

[17]. Український переклад документації СУБД MySQL, виконаний в 2016 р. компанією Ensita.NET.

ДОДАТОК А. Відомість матеріалів кваліфікаційної роботи

№	Формат	Найменування	Кількість листів	Примітка
1	А4	Реферат	2	
2	А4	Список умовних скорочень	1	
3	А4	Зміст	1	
4	А4	Вступ	2	
5	А4	Стан питання. Постановка задачі	20	
6	А4	Спеціальна частина	51	
7	А4	Економічний розділ	11	
8	А4	Висновки	1	
9	А4	Перелік посилань	1	
10	А4	Додаток А	1	
11	А4	Додаток Б	1	
12	А4	Додаток В	1	
13	А4	Додаток Г	1	

ДОДАТОК Б. Перелік файлів на електронному носії

1. Пояснювальна Записка.docx
2. Кваліфікаційна робота Олефір К.А..docx
3. Презентація.pttx
4. ОлефірКА-125м-22-1-КР.pdf (3).p7s
5. ОлефірКА-125м-22-1-КР.pdf (3).p7s.xml
6. ОлефірКА-125м-22-1-КР.pdf (3)_Validation_Report.pdf

ДОДАТОК В. Відгук керівника економічного розділу

Економічний розділ виконаний відповідно до вимог, які ставляться до кваліфікаційних робіт, та заслуговує на оцінку 90 б. («дев'яносто»).

Керівник розділу _____ доц. Пілова Д.П.

(підпис) (ініціали, прізвище)

ДОДАТОК Г. Відгук керівника кваліфікаційної роботи.

В І Д Г У К

на кваліфікаційну роботу студента групи 125м-22-1

Олефіра Кирила Андрійовича

на тему: «Засоби контролю інформаційної безпеки в базах даних SQL»

Пояснювальна записка ст. Олефіра Кирила Андрійовича складається зі вступу, трьох розділів і висновків, викладених на 96 сторінках. Кваліфікаційна робота присвячена актуальній темі забезпечення інформаційної безпеки в базах даних SQL. Вона охоплює аналіз потенційних загроз безпеці даних у системах MySQL, Oracle, та MS SQL Server. Робота також розглядає методи та засоби контролю інформаційної безпеки у контексті поліпшення захисту даних.

Тема кваліфікаційної роботи безпосередньо пов'язана з об'єктом діяльності магістра спеціальності 125 «Кібербезпека». "Для досягнення мети цієї кваліфікаційної роботи вирішуються наступні завдання: аналіз загроз інформаційній безпеці в SQL базах даних; дослідження методів та засобів контролю інформаційної безпеки; розробка стратегій ефективного захисту даних.

Практичне значення цієї роботи полягає у визначенні ключових загроз інформаційній безпеці у SQL базах даних та розробці методів та засобів для ефективного контролю інформаційної безпеки.

Кваліфікаційна робота виконана відповідно до вимог і може бути допущена до захисту, оцінка – 90 (відмінно). Олефір К.А. заслуговує присвоєння звання магістра з кібербезпеки за спеціальністю 125 Кібербезпека, освітня програма «Кібербезпека».

Рівень запозичень у кваліфікаційній роботі не перевищує вимог «Положення про систему виявлення та запобігання плагіату».

Керівник кваліфікаційної роботи

Керівник спец. розділу